



IPG Politécnico
|da|Guarda
Polytechnic
of Guarda

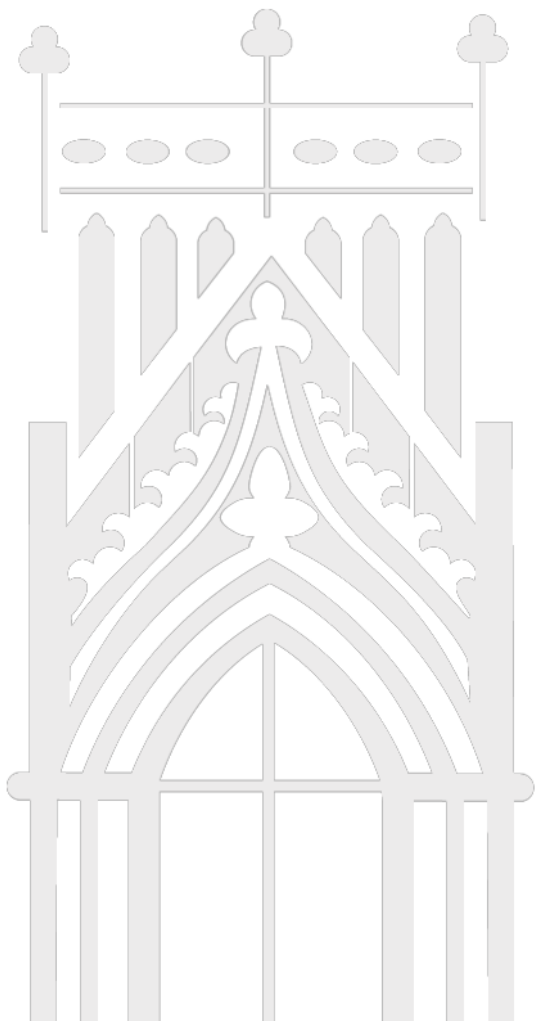
Mestrado em Computação Móvel

Workpresence

Aplicação para a gestão de presenças

Vasco Manuel de Deus Lello Fortuna

julho | 2019



Escola Superior
de Tecnologia e Gestão



Workpresence

Aplicação para a gestão de presenças

Relatório de Projeto Aplicado submetido como requisito parcial para
obtenção do grau de Mestre em Computação Móvel

Orientadora: Professora Doutora Natália Gomes

Vasco Manuel de Deus Lello Fortuna

Abril 2019

“Para dominar a tecnologia, tu tens de brincar com ela.”

Jordan B. Peterson

AGRADECIMENTOS

Gostaria de agradecer há minha família toda, especialmente aos meus pais (Manuel Fortuna e Rosa Lello), cujo apoio e encorajamento foram indispensáveis para a concretização deste trabalho. Não existem palavras que definam a enorme gratidão que tenho por eles.

Ao meu amigo que esteve mais perto: David Tavares e aos que estiveram mais longe: Hélio Cruz, Igor André e Yashar Sabaz, pela motivação e pelos bons momentos que me deram ao longo deste tempo todo.

RESUMO

A gestão e controlo de presenças nas organizações tem, ao longo dos anos, vindo a merecer uma atenção cada vez mais significativa no meio das organizações. Existe uma perceção clara que o controlo eficaz e análise de presenças uma maior produtividade dos funcionários e, consequentemente, das organizações.

O presente trabalho consiste no processo de desenvolvimento de um sistema informático para a gestão de presenças de funcionários de uma organização, na qual os funcionários utilizem uma aplicação móvel no seu smartphone, para registarem as suas entradas e as suas saídas. O administrador da organização terá acesso às suas ferramentas de gestão, visualização de estatísticas e à atividade dos seus funcionários através de uma aplicação Web. Estas duas aplicações estabelecerão comunicação entre as duas, através de tecnologia WebService.

Já existem algumas aplicações no mercado de gestão de presenças, no entanto a grande maioria faz uso de sensores físicos, para controlar as presenças dos funcionários. Estes sensores físicos trazem algumas desvantagens como: requerem amplo tempo a serem instaladas, os funcionários necessitam de adaptar a novas rotinas, estão sujeitos a manutenções periódicas e em caso de avaria, a sua reparação pode demorar semanas. Todas estas desvantagens fazem com que os sensores físicos sejam bastante invasivos e difíceis de se adaptar a mudanças dentro de uma organização.

O sistema proposto, *designa-se* por WorkPresence, utiliza as tecnologias Android e Web para disponibilizar a qualquer organização, um sistema de gestão de presenças que seja adaptável, intuitivo, pouco invasivo, sem custos de manutenção e utilizável em qualquer computador ou smartphone Android.

Palavras-chave: *Android, Web Services, Gestão de presenças, Geolocalização*

ABSTRACT

Through recent years, the management and control of presences of organizations has earned the attention of their administrators. There is a clear perception that the effective control and analysis of presences improves the productivity of the employees and, consequently, their organizations.

This project consists in the process of the development of an informatics system, with the objective of managing the employee's presences of an organization. In which, the employees use a mobile app to register their check-ins and checkouts. The administrator of the organization will have access to its management tools, employee's activity and statistics, through a Web app. These two apps will establish connection through WebService technology.

There are already a few applications being sold commercially. However, the majority makes use of physical sensors to control the employee's presences. These sensors bring along some disadvantages: they require a lengthy installation process, the employees need to adapt their daily routines, the sensors require periodic maintenance and sensor's failure or malfunction can take a long time to be repaired.

The proposed system goes by the name of WorkPresence. It makes use of Android and Web technology to provide, to any organization, a management system that is adaptable, intuitive, non-intrusive, with no maintenance costs and usable on any computer or Android smartphone.

Keywords Android, Web Services, Presence management, Geolocation

Índice

Agradecimentos.....	i
Resumo.....	ii
Abstract.....	iii
Índice de Figuras	vi
Índice de Tabelas.....	vii
Lista de abreviaturas.....	viii
1. INTRODUÇÃO.....	1
1.1. Contextualização.....	1
1.2. Objetivos propostos	2
1.2.1. Aplicação Móvel.....	2
1.2.2. Aplicação Web	3
1.3. Estrutura.....	3
2. Sistemas de apoio à gestão de presenças	4
2.1. A importância da gestão de presenças de funcionários	4
2.2. Problemas dos sistemas biométricos.....	6
2.2.1. Implicações sociais e éticas de identificação biométrica.....	7
2.3. Proteção de Dados	7
2.3.1. Artigo 35 da CRP – utilização da informática.....	8
2.3.2. Encarregado de proteção de dados	8
2.3.3. Direito ao esquecimento	8
2.3.4. Segurança na proteção de dados.....	9
2.4. Análise de aplicações existentes no mercado	10
2.4.1. CucoCloud.....	10
2.4.2. Attendance Manager [8]	13
2.4.3. SynchroTeam [9]	14
3. Planeamento do projeto	19
3.1. Metodologia	19
3.1.1. Extreme Programming.....	20
3.2. Etapas do projeto	22
3.3. Análise de requisitos	23
3.3.1. Requisitos funcionais.....	23
3.3.2. Requisitos não-funcionais	24
3.4. Modelação UML.....	25
3.4.1. Diagrama de contexto.....	25
3.4.2. Diagrama de casos de uso.....	27
3.4.3. Descrições de caso de uso/Diagramas de sequência	27
3.4.4. Modelo ER e semântica de dados.....	35
3.4.5. Diagrama de classes.....	41
4. Implementação.....	43
4.1. Tecnologias utilizadas no projeto	43
4.1.1. HTML.....	43

4.1.2.	CSS e Bootstrap.....	44
4.1.3.	Javascript, AJAX e jQuery	45
4.1.4.	PHP.....	46
4.1.5.	MySQL.....	47
4.1.6.	Android.....	48
4.1.7.	Web Services	49
4.2.	Desenvolvimento	50
4.2.1.	Aplicação Web	50
4.3.	Aplicação móvel	62
4.3.1.	Diagrama de Hierarquia.....	62
4.3.2.	Navegação entre interfaces.....	63
4.3.3.	Interligações do sistema.....	65
4.4.	Testes	67
4.4.1.	Cálculo da distância entre pontos geográficos	67
5.	Conclusões.....	70
6.	Referências Bibliográficas.....	73
Anexos.....		75
ANEXO A1		76
ANEXO A2		77
ANEXO A3		78
ANEXO B1		79
ANEXO C1		80
ANEXO D1		81
Diagramas de casos de uso e sequência		81

ÍNDICE DE FIGURAS

Figura 1 - Interface "Picar" da CucoCloud.....	11
Figura 2 - Interface "Relatório" da CucoCloud	11
Figura 3 - Interface "Perfil" da CucoCloud	12
Figura 4 - Menu principal do Attendance Manager	13
Figura 5 - Aplicação Web da SynchroTeam	15
Figura 6 - Menu principal da aplicação móvel da SynchroTeam	16
Figura 7 - Mapa de Gantt previsto.....	22
Figura 8 - Diagrama de contexto do administrador.....	26
Figura 9 - Diagrama de contexto do funcionário.....	26
Figura 10 - Diagrama de Casos de Uso do Workpresence	27
Figura 11 - Diagrama de sequência do "Check-in"	29
Figura 12 - Diagrama de sequência "Ver localização"	30
Figura 13 - Diagrama de Sequência "Ver estatísticas do funcionário"	31
Figura 14 - Diagrama de sequência "Gerir Funcionários"	33
Figura 15 - Diagrama de sequência "Ver Atividade de Funcionário"	34
Figura 16 - Modelo ER.....	35
Figura 17 - Operações da tabela "PontosOrganizações"	40
Figura 18 - Diagrama de classes.....	41
Figura 19 - Diagrama de hierarquia da aplicação web	52
Figura 20 - Interface "Atividade Funcionários"	55
Figura 21 - Exemplo da função "FillTable"	61
Figura 22 - Diagrama de hierarquia da aplicação móvel.....	62
Figura 23 - Interface "Marcação de presenças"	63
Figura 24 - Diferenças entre Activities e Fragments [29]	64
Figura 25 - Ligações entre os vários componentes do sistema.....	65
Figura 26 - Resultado do primeiro teste da função "VerDistanciaPontos"	68
Figura 27 - Resultado do segundo teste da função "VerDistanciaPontos"	69
Figura 28 - Diagrama de sequência do "Checkout"	82
Figura 29 - Diagrama de sequência "Ver Perfil"	83
Figura 30 - Diagrama de sequência "Gerir Horários"	85
Figura 31 - Diagrama de sequência "Colocar Funcionário em Horário"	86

ÍNDICE DE TABELAS

Tabela 1 - Comparação das aplicações estudadas (S:Sim, N:Não)	18
Tabela 2 - Descrição de caso de uso "Check-in"	28
Tabela 3 - Descrição de caso de uso "Ver Localização"	30
Tabela 4 - Descrição de caso de uso "Ver estatísticas de funcionário"	31
Tabela 5 - Descrição de caso de uso "Gerir Funcionários"	32
Tabela 6 - Descrição de caso de uso "Ver atividade de funcionário"	34
Tabela 7 - Dicionário de dados da tabela "Organizações"	36
Tabela 8 - Operações da tabela "Organizações"	36
Tabela 9 - Dicionário de dados da tabela "Funcionários"	37
Tabela 10 - Operações da tabela "Funcionários"	38
Tabela 11 - Dicionário de dados da tabela "Sessões"	38
Tabela 12 - Operações da tabela "Sessões"	38
Tabela 13 - Dicionário de dados da tabela "Departamentos"	39
Tabela 14- Dicionário de dados da tabela "Horários"	39
Tabela 15 - Operações da tabela "Horários"	39
Tabela 16 - Descrição de caso de uso "Checkout"	81
Tabela 17 - Descrição de caso de uso "Ver perfil"	83
Tabela 18 - Descrição de caso de uso "Gerir Horários"	84
Tabela 19 - Descrição de caso de uso "Colocar funcionários em horário"	86

LISTA DE ABREVIATURAS

CSS – *Cascading Style Sheets*

CNPD – Comissão Nacional de Proteção de Dados

CRP – Constituição da República Portuguesa

GPS – *Global Positioning System*

HTML - *HyperText Markup Language*

IDE – *Integrated Development Enviroment*

PHP – *Hypertext Preprocessor*

RGPD – Regulamento Geral de Proteção de Dados

SDK – *Software Development Kit*

UML – *Unified Modelling Language*

1. Introdução

O presente relatório caracteriza e detalha a análise e desenvolvimento do projeto, chamado WorkPresence, realizado pelo aluno Vasco Manuel de Deus Lello Fortuna, no âmbito da tese de Mestrado de Computação Móvel, da Escola Superior de Tecnologia e Gestão do Instituto Politécnico da Guarda.

1.1. Contextualização

A constante evolução, no que diz respeito às tecnologias de informação e comunicação, veio influenciar e melhorar vários aspetos do nosso quotidiano e em particular da sociedade, especificamente, de tarefas repetitivas que consomem bastante tempo e são pouco eficientes. Uma dessas tarefas é a gestão de presenças de funcionários. Esta gestão, essencial para qualquer organização, consegue-se tornar muito fastidiosa e pode levar a erros contínuos se não for feita corretamente e rigorosamente.

Existe, atualmente, no mercado das aplicações, alguma variedade de ferramentas/aplicações online que têm por objetivo fornecer esta gestão de presença aos membros das organizações. No entanto, muitas destas ferramentas já se encontram desatualizadas e/ou não tomam proveito total de todas as potencialidades que as tecnologias atuais oferecem permitindo melhorar a produtividade das organizações no seu todo e em particular na gestão dos seus recursos humanos.

Para além das aplicações online, também existem soluções que recorrem, como é exemplo, a sensores externos (RFID, leitor de impressão digital, leitor de íris, entre outros) para identificar os trabalhadores e registar a sua presença nas organizações.

De modo a alcançar os objetivos propostos, considerou-se desenvolver um sistema composto por dois componentes: uma aplicação móvel para os recursos humanos, funcionários de uma organização, e uma aplicação web para os administradores. A aplicação web fará uso de Web Services que permitirão interligar as duas componentes.

A análise de mercado destas soluções permitiu-nos verificar que a maioria das soluções existentes (especialmente as que utilizam sensores biométricos) requerem um contacto direto com os fornecedores do produto, para o produto ser ajustado às

necessidades e características da organização. A utilização desta solução obriga a que os sensores ou sistemas sejam fisicamente instalados, que as organizações aprendam a utilizar novos recursos informáticos, produtos estes que requerem uma manutenção local periódica.

Tendo em vista as novas potencialidades e soluções oferecidas pelas tecnologias atuais e considerando que a utilização de sistemas de registo biométrico pode ser um processo demorado, que envolve alguns custos significativos e pode causar problemas nas organizações. Pretende-se com este projeto desenvolver um sistema informático, que permita efetuar o registo e controlo de presenças nas organizações de forma acessível, fácil, intuitivo e gratuito.

O sistema, tal como referido, é composto por dois subsistemas. O primeiro é uma aplicação web, que tem por objetivo o acesso generalizado pelos seus utilizadores em qualquer dispositivo, controlar e analisar as entradas e saídas dentro da organização. A interface desta aplicação tem como requisitos ser um sistema intuitivo, automatizado e de fácil uso e rápida aprendizagem de utilização.

O segundo subsistema, de registo de presenças, é a aplicação móvel que tem como objetivo substituir qualquer tipo de sensor, registando os funcionários da organização através da sua localização. É de salientar, que a informação relativa ao registo e controlo de entradas e saídas na organização, estatísticas, encontra-se disponível para o Administrador do sistema e para o respetivo funcionário.

1.2. Objetivos propostos

A partir da contextualização apresentada conseguimos definir vários objetivos que o projeto deve cumprir. De acordo com os objetivos gerais, definidos para cada sistema, expomos os seus requisitos principais:

1.2.1. Aplicação Móvel

O sistema incorporará uma aplicação móvel dedicada ao registo de presenças dos funcionários, onde cada funcionário individual poderá marcar as suas presenças e visualizar o seu histórico de presenças. Assim, podemos determinar as seguintes funcionalidades, necessárias ao correto funcionamento do sistema:

- Registo de login dos funcionários.
- Registo de check-in/check-out do funcionário.
- Visualizar horário de trabalho do funcionário.
- Analisar estatísticas referentes às horas de trabalho do funcionário.

1.2.2. Aplicação Web

O sistema também integrará uma aplicação *Web* estará focalizada na gestão de funcionários, por parte dos administradores de cada organização. A partir desta aplicação, cada administrador poderá visualizar a atividade dos seus funcionários, gerir os registos e contas de cada funcionário e visualizar as estatísticas de horas de trabalho de cada funcionário. Assim, obtemos os seguintes objetivos para a aplicação *Web*:

- Gerir funcionários e respetivos departamentos afetos aos funcionários.
- Visualizar presenças dos funcionários.
- Gerir horários de trabalho para os funcionários.
- Analisar estatísticas de trabalho de cada funcionário e departamento.

1.3. Estrutura

O presente projeto aplicado encontra-se dividido em cinco capítulos. No primeiro capítulo, são apresentados os objetivos do projeto, bem como uma breve contextualização do mesmo. No segundo capítulo, é apresentado o estado de arte, contendo um estudo às aplicações existentes que se enquadram no tema do projeto. No terceiro capítulo, é apresentada a metodologia e a análise de requisitos. No quarto capítulo, as tecnologias utilizadas são descritas, tal como algumas decisões importantes que foram tomadas durante o desenvolvimento. No quinto, e último capítulo, é apresentada a conclusão do projeto e algumas perspetivas futuras de melhoria de desenvolvimento da aplicação.

2. SISTEMAS DE APOIO À GESTÃO DE PRESENÇAS

Este capítulo tem como objetivo compreender o atual estado da arte, com recurso à análise de artigos científicos, da gestão e controlo de presenças de recursos humanos nas organizações, e compreender o funcionamento das soluções informáticas existentes no mercado com o objetivo de analisar, entender e definir os possíveis requisitos do projeto.

2.1. A importância da gestão de presenças de funcionários

De modo a perceber melhor a área de estudo do projeto, e a sua importância para as organizações, apresentamos de forma sucinta, através da análise de diferentes artigos disponíveis online, o porquê e a necessidade da gestão de presenças para as organizações.

O artigo “The Importance of Employee Attendance Reporting” detalha os porquês da existência de sistemas de gestão de presenças e o seu valor dentro das organizações.

Segundo este artigo, [1], as presenças dos funcionários têm impacto na: organização dos recursos humanos, gestão de horários de trabalho, formação dos indivíduos, produtividade e moral da organização. A recolha das horas dos trabalhadores serve, assim, para compreender as necessidades laborais, tendências de férias, bem como registo de dados importantes para o correto funcionamento da organização de acordo com a legislação em vigor.

A recolha destes dados permite, entre outros, determinar se a organização se encontra a par dos seus objetivos de modo manter-se competitiva, tendo em vista o bem-estar do trabalhador. Ao monitorizar as presenças dos funcionários, as organizações conseguem distinguir os empregados que chegam cedo, a tempo ou constantemente tarde, ou ainda os que recorrentemente fazem horas extras. Este tipo de informação pode nalguns casos, ajudar a decidir quais são os empregados que são uma mais-valia para a

organização e/ou quais os departamentos deficitários em termos de recursos humanos. O acesso a este tipo de informação permite que a organização melhore a sua produtividade e eficiência.

No artigo “The Importance of Employee Time Tracking”, são explicadas as diversas vantagens do controlo das horas de trabalho dos funcionários. [2] Deste artigo, são destacadas as seguintes vantagens:

1. Aumento do valor da organização a potenciais clientes. O controlo do tempo de trabalho dos funcionários estabelece credibilidade como uma organização que valoriza tempo e precisão. *Software* que controla o tempo de trabalho dos funcionários, também contém informações detalhadas de cada funcionário e os seus padrões de trabalho, sendo um grande ponto de venda para qualquer empresa que queira comprar a organização.
2. Melhoramento da moral dos funcionários. Um ponto muito subvalorizado é o facto de o controlo de horas de trabalho reforçar os sentimentos de confiança e respeito dos funcionários, aumentando a sua produtividade e performance. Quando uma organização não tem um sistema de controlo implementado, existe uma grande possibilidade de os funcionários estarem em constante inspeção, o que pode causar imenso stress e medo de repercussões por qualquer erro que façam.

No que diz respeito aos sistemas informáticos, de gestão de presenças, estes possibilitam, aos administradores de sistemas, uma maior eficiência na gestão de recursos humanos, através da análise de estatística e produção de relatórios. Ao recolher dados de diferentes fontes de informação (Web check-in/out, geolocalização e horas e datas dos smartphones), praticamente garantimos a integridade da informação das presenças dos funcionários. Assim, eliminamos a duplicação e erros de registos dos métodos tradicionais de recolha de presenças. Tal como referido, a análise de estatísticas e padrões das horas de trabalho, providencia ao departamento de recursos humanos, informação útil para autorizar pedidos de ausência de trabalho, fazer projecções de recrutamento de pessoal e horários de turnos.

Ao longo do tempo, vários foram os sistemas desenvolvidos que permitem registar e automatizar as presenças. Um dos métodos que se tornou popular foi os sistemas biométricos, especialmente os que são constituídos com sensores de leitura de impressão digital. Não obstante, estes sistemas apresentam alguns problemas que passamos a descrever.

2.2. Problemas dos sistemas biométricos

Ainda que comumente utilizados pelas organizações os sistemas biométricos apresentam alguns problemas que foram identificados, e aqui apresentamos, na dissertação de mestrado de Elsa Araújo com o título RUN [3]. Ao longo desta dissertação, encontramos algumas problemáticas que surgiram ao introduzir um sistema biométrico num hospital. Dos problemas identificados salientamos os seguintes, que nos permitiram justificar algumas decisões deste projeto:

- As máquinas biométricas desgastam-se com o tempo, como exemplo, botões dos algarismos danificados.
- Se ocorrerem erros de *software* (ex: atualizações ou configurações demoradas), é necessário deslocar-se até outro sistema biométrico para marcar presença.
- Segundo os funcionários do hospital, a assistência às máquinas, pelos técnicos, não é imediata e pode até levar semanas.
- Frequentemente existem ocorrências em que o sistema biométrico não reconhecia as impressões digitais, de certos profissionais, pelo que as presenças eram assinaladas em papel.
- A implementação do sistema levou a dificuldades de adaptação e dificuldades de integração.

É de notar que, apesar destes obstáculos todos, a autora da dissertação concluiu que o sistema biométrico implementado não foi em vão. Justificando que criou uma cultura de responsabilidade com a organização [3].

2.2.1. Implicações sociais e éticas de identificação biométrica

O artigo de nome “Ethical and social implications of biometric identification technology”, escrito por Emilio Moridini e Carlo Petrini, em 2017, discute os aspetos sociais e éticos dos sistemas biométricos.

De forma resumida, o artigo questiona se a identificação biométrica constitui uma invasão à privacidade pessoal e se existe potencial de se tornar uma arma nas mãos de governos autoritários. À medida que identificadores biométricos se tornam mais universais, eles podem revelar mais do que a identidade de uma pessoa, comprometendo a nossa privacidade de uma maneira mais meticulosa. O artigo conclui com a pergunta se estamos prontos para termos partes do corpo (dedos, olhos e fala) guardadas em bases de dados e trocadas como comodidades por organizações. Muitas destas questões sociais e éticas deveriam ser sumarizadas num debate sobre dignidade humana [4].

2.3. Proteção de Dados

Tendo em consideração alguns dos problemas identificado com os sistemas biométricos e uma vez que um sistema de informação tem como pressuposto o registo de dados consideramos importante, tendo em vista os objetivos do projeto, ter conhecimento da legislação em vigor de registo e proteção de dados. Assim, e de forma sucinta, expomos algumas questões relativas ao Regulamento Geral Proteção de Dados (RGPD) que pode afetar o nosso projeto.

Sendo que um dos objetivos do projeto se prende com o registo de dados de recursos humanos de uma organização (ex. funcionários e/ou alunos), é importante ter em conta a legislação nacional que protege os dados pessoais dos seus utilizadores. A presente legislação encontra-se no site oficial do RGPD¹. De acordo com este regulamento, apresentamos alguns artigos, que obrigam a que o presente projeto fosse modelado e implementado de diferente maneira, isto é, tendo em vista o cumprimento do RGPD [5].

¹ <https://protecao-dados.pt/o-regulamento/>

2.3.1. Artigo 35 da CRP – utilização da informática

- 1) Todos os cidadãos têm o direito de aceder aos dados informatizados que lhe digam respeito. De acordo com este pressuposto, os funcionários irão apenas ter acesso aos seus dados pessoais e suas estatísticas através da aplicação móvel.
- 2) A informática não pode ser utilizada para tratamento de dados referentes a convicções filosóficas ou políticas, filiação partidária ou sindical, fé religiosa, vida privada e origem étnica. Isto significa que não a base de dados não poderá armazenar dados relevantes, tais como opiniões e estilos dos funcionários (políticas, religiosas, etc...). A base de dados deverá restringir o registo a dados essenciais para a gestão de presenças na organização.
- 3) É proibido o acesso a dados pessoais de terceiros, salvo em casos excepcionais previstos na lei. Só os administradores do sistema é que pode ter acesso aos dados dos recursos em causa (ex. funcionários, alunos).

2.3.2. Encarregado de proteção de dados

O regulamento indica que tem de existir um “Encarregado de proteção de dados” sempre que o tratamento de dados for efetuado por uma autoridade ou um organismo público. No contexto deste projeto, o autor vai tomar a posição de encarregado, garantido que o sistema trate os dados pessoais de cada funcionário licitamente.

2.3.3. Direito ao esquecimento

Os utilizadores têm direito à eliminação dos dados ("direito a ser esquecido"), significando que o titular dos dados tem o direito de solicitar a eliminação dos seus dados pessoais, sem demora injustificada [6]. Ou seja, o sistema tem de estar preparado para eliminar funcionários e todos os seus dados pessoais.

2.3.4. Segurança na proteção de dados

O Regulamento obriga, ao responsável pelo tratamento de dados e ao subcontratante, que as organizações apliquem medidas que assegurem um nível de segurança de dados apropriados.

Estas medidas incluem:

1. Utilização de pseudónimos e a cifragem dos dados pessoais;
2. A capacidade de assegurar a confidencialidade, integridade, disponibilidade e resiliência permanentes dos sistemas e dos serviços de tratamento;
3. A capacidade de restabelecer a disponibilidade e o acesso aos dados pessoais de forma atempada no caso de um incidente físico ou técnico [6].

2.4. Análise de aplicações existentes no mercado

A presente secção tem como objetivo apresentar aplicações cujas funcionalidades se enquadram na área deste projeto, analisar os seus benefícios informáticos, as suas limitações e analisar se atualmente existe alguma aplicação que se assemelhe aos objetivos do projeto. Pretende-se também, obter informações e conhecimentos sobre aspetos que poderão ser benéficos para o desenvolvimento do projeto.

Esta análise e levantamento de recolha de dados tem como objetivo perceber o tipo de requisitos que este *software*, gestão de presenças, deve ter. Das inúmeras soluções disponíveis no mercado estudadas, e face aos objetivos, apresentamos uma análise das seguintes aplicações: CucoCloud, Attendance Manager e a SynchroTeam.

2.4.1. CucoCloud

Após análise detalhada à aplicação, portuguesa CucoCloud, verificámos que esta tem objetivos bastante idênticos aos definidos para este projeto. É um sistema de gestão e controlo de assiduidade desenvolvido na Cloud, com administração através de uma aplicação web, em que os funcionários podem marcar presença através de uma aplicação para dispositivos móveis. Adicionalmente, a CucoCloud também disponibiliza um sistema biométrico, através de impressão digital, para controlo de entradas, com custos adicionais.

A CucoCloud teve o seu foco principal no desenvolvimento para *tablets*, chamada CucoTablet, que não foi estudada por não corresponder com os objetivos principais do projeto, especificamente, porque a aplicação do projeto será desenvolvida para uso pessoal e não partilhado por vários utilizadores. Segundo o website oficial², o CucoTablet substitui um sistema biométrico, onde os colaboradores introduzem um pin na aplicação de um tablet (supostamente possuído pela empresa), que marca a entrada do colaborador na organização [7].

A aplicação móvel da CucoCloud tem um *design* simples e uma interface intuitiva, como podemos ver na Figura 1. Tem três botões com as funcionalidades essenciais: Picar, Relatórios e Perfil.

² <https://cucocloud.pt/>

A interface “Picar”, permite ao utilizador marcar a sua presença remotamente e registar a sua localização e a hora de entrada. Também mostra as picagens do utilizador mais recentes. No entanto, ao testar a aplicação, verificou-se que não conseguia detetar o GPS do *smartphone*, impossibilitando o teste desta funcionalidade.

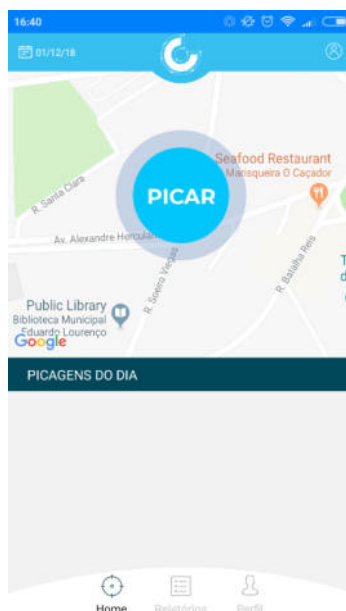


Figura 1 - Interface "Picar" da CucoCloud

A interface “Relatório”, Figura 2, apresenta todas as picagens feitas pelo utilizador. Mostra as horas totais e a média de horas de cada picagem. O utilizador pode ainda escolher entre as picagens do dia, semana, mês ou datas específicas.

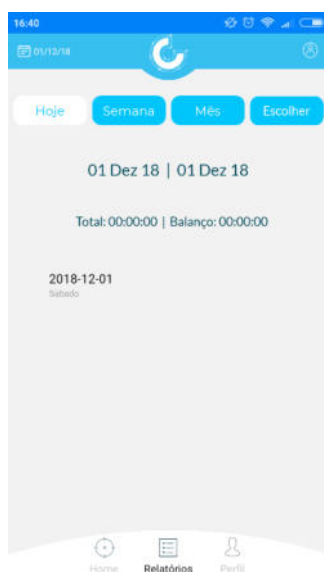


Figura 2 - Interface "Relatório" da CucoCloud

A interface “Perfil”, Figura 3, apresenta os dados de perfil do utilizador (foto de perfil, nome da empresa, nome e email) e permite fazer *logout* da aplicação.

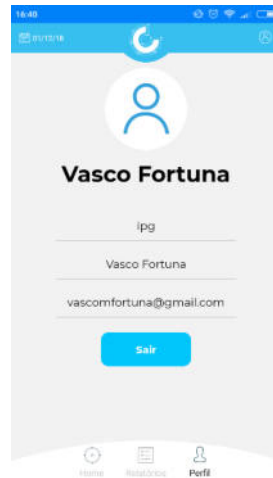


Figura 3 - Interface "Perfil" da CucoCloud

Para além do sistema em Cloud que providenciam, também fornecem um sistema biométrico que custa um mínimo de 199€. A este preço acresce um custo adicional de 10€/mensais para que seja possível tratar os dados na aplicação Web. O equipamento deste sistema é composto por uma única máquina que recolhe impressões digitais. Ao colocar o dedo no aparelho, é marcada a picagem do utilizador e os dados são enviados para a Cloud.

2.4.2. Attendance Manager [8]

Attendance é uma aplicação móvel gratuita dirigida a estudantes. A aplicação é destinada à utilização individual do estudante, tendo só interatividade com um único dispositivo. Após o estudo da aplicação, foram retiradas as principais funcionalidades e características relevantes para este projeto:

- 1) Criação e gestão de horários semanais ou palestras.
- 2) Regista as presenças ou faltas do aluno.
- 3) Visualização de estatísticas das aulas presenciadas.
- 4) *Design* simples e intuitivo (Figura 4).

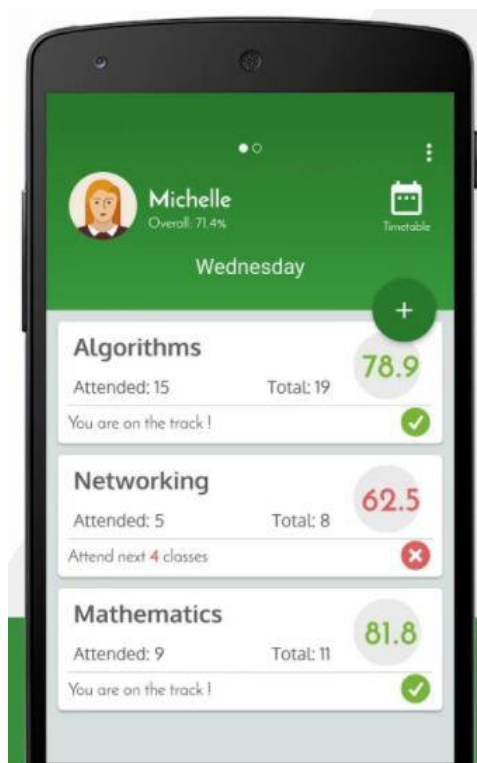


Figura 4 - Menu principal do Attendance Manager

Uma das grandes vantagens do Attendance é o seu *design* simples e intuitivo, encontrando-se toda a informação essencial numa única interface, fácil de compreender e visualizar. Este *design*, possivelmente, servirá de inspiração para o *design* da aplicação móvel do projeto a desenvolver.

Apesar das interfaces da aplicação Attendance apresentarem um aspeto simples, o mesmo não nos oferece características que irão inspirar ou auxiliar os objetivos do nosso projeto. A isto se deve o facto de que o sistema Attendance foi desenvolvido na perspetiva de ser uma aplicação dedicada a apenas um utilizador (sem interação com outros utilizadores). Também, por ser feita especificamente para estudantes, não conseguimos retirar ideias sobre como gerir os horários.

2.4.3. SynchroTeam [9]

SynchroTeam é um produto pago e que tem como objetivo a gestão de presenças de funcionários. Este produto encontra-se dividido em duas partes: uma aplicação móvel para os funcionários e uma aplicação web para os administradores, que vai de encontro aos nossos objetivos. O administrador cria tarefas, atribuí-lhes um prazo ou um horário e pode *designar* funcionários específicos para cumprirem essa tarefa. Estas tarefas são equivalentes a pedidos de clientes, contendo campos de faturação e horas de trabalho necessárias.

A aplicação Web, SynchroTeam, tem as seguintes características:

- 1) Criação e gestão de tarefas para os funcionários.
- 2) Mapa com a localização dos funcionários.
- 3) Área de clientes, contendo o histórico e contacto.
- 4) Atribuição de trabalhos a funcionários.
- 5) Área de feedback dos funcionários.

A aplicação móvel contém as seguintes características:

- 1) Lista de tarefas a efetuar.
- 2) Descrição de tarefas.
- 3) Área de faturação.
- 4) Notificações de novas tarefas.

A SynchroTeam contém funcionalidades idênticas aos objetivos deste projeto, incluindo a ligação entre uma aplicação web e uma aplicação móvel, permite a gestão de localizações e o registo de horas de trabalho.

No entanto, como tem criação de tarefas em vez de horários semanais, é uma aplicação mais que tem em vista a completar projetos encomendados por clientes e não é dedicada a organizações com horários regulares, que não estão dependentes de pedidos de clientes. Esta funcionalidade também não permite gerir muitos trabalhadores eficientemente, porque cada tarefa tem de ser atribuída individualmente a cada funcionário. Esta característica não permite e dificulta, na nossa opinião, gerir organizações com um grande número de funcionários. Como podemos observar na Figura 5, este nível de detalhe é excessivo para os nossos objetivos.

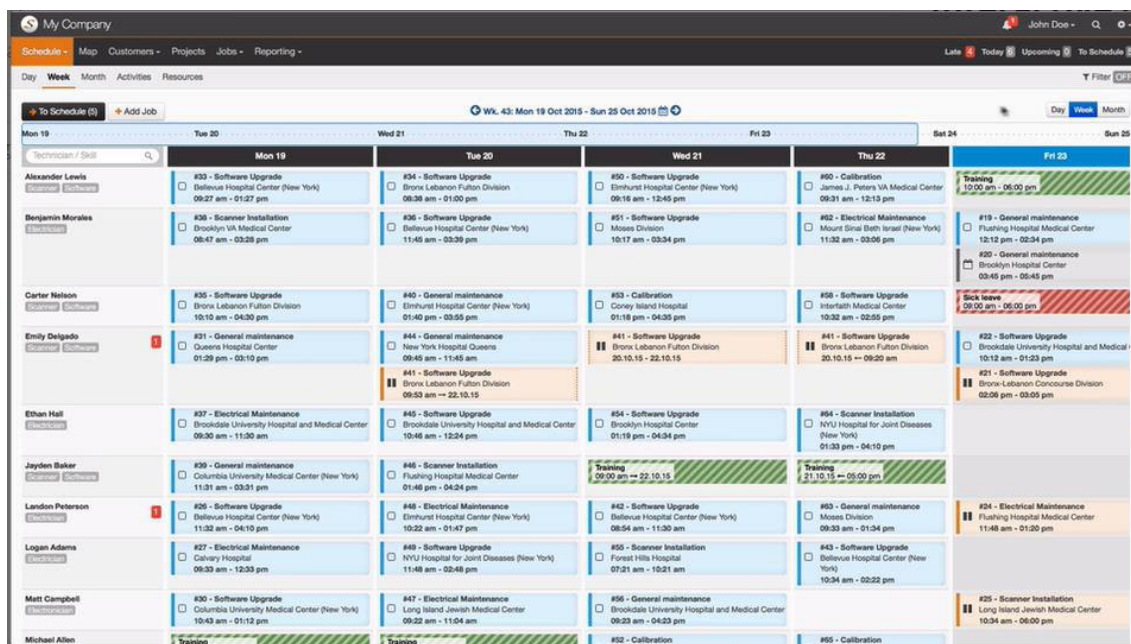


Figura 5 - Aplicação Web da SynchroTeam

A aplicação mobile, SynchroTeam, destinada a funcionários de uma organização, permite a que os mesmos possam aceder a informação necessária sobre as suas tarefas, os horários, a faturação, o requerimento de equipamento e relatórios, entre outras, Figura 6.

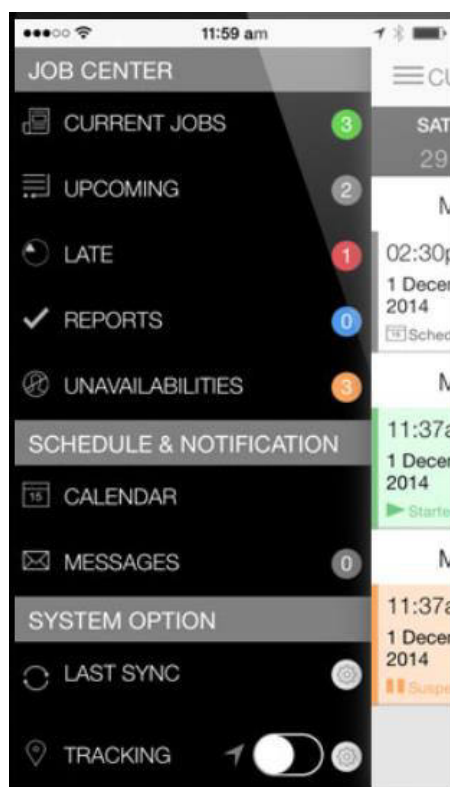


Figura 6 – Menu principal da aplicação móvel da SynchroTeam

Não foi possível testar a aplicação mobile porque o fornecedor não fornece a aplicação mobile de forma gratuita, ou em modo Trial, que consideramos ser um erro em termos de marketing digital. Não obstante, foi possível retirar algumas conclusões sobre esta aplicação, a partir da análise do seu site oficial³:

1. A conjugação de tantas funcionalidades numa só aplicação, torna-a pouco intuitiva, o que pode levar algum tempo por parte da organização a compreender o seu funcionamento e oferecer formação aos funcionários.
2. A aplicação regista a localização em tempo real, o qual não é permitido, na Europa, de acordo com o RGPD.
3. Não existe registo de presenças na aplicação móvel. O funcionário só pode usar a aplicação para dizer que completou as tarefas que lhe foram atribuídas pelo administrador.

³ <https://www.synchroteam.com/mobile.php>

Todas estas características mostram que SynchroTeam, apesar de integrar uma ligação entre aplicação móvel e web, não é o produto ideal para a concretização do nosso projeto. Além de não possibilitar a gestão de presenças, a gestão de tarefas é complexa, objetivo contrário a este projeto: ter uma aplicação intuitiva e fácil de integrar nas organizações.

No entanto, a análise permitiu-nos retirar alguns aspetos que deveremos evitar aquando da implementação no nosso projeto, WorkPresence:

- Evitar a aglomeração de muitas funcionalidades numa só aplicação ou numa só interface.
- Minimizar a quantidade de informação numa única interface, para que esta seja visualmente apelativa e de fácil compreensão para o utilizador.

A análise destas três aplicações, CucoCloud, Attendance Manager e SynchroTeam, permitiu-nos definir uma série de requisitos, que consideramos fundamentais para o projeto WorkPresence, de modo a que este seja relativamente à concorrência, um projeto mais eficiente de acordo com as necessidades atuais das organizações, como podemos ver na tabela seguinte (Tabela 1):

	<i>CucoCloud</i>	<i>Attendance M.</i>	<i>SynchroTeam</i>
Aplicação Web	S	N	S
Aplicação Móvel	S	S	S
Gestão de funcionários	S	N	S
Registo de presenças	S	S	N
Estatísticas de presenças	S	S	S
Interface intuitivo	S	S	N
Utilização de sistemas biométricos	S	N	N
Gratuito	N	S	N

Tabela 1 - Comparação das aplicações estudadas (S:Sim, N:Não)

Assim, conclui-se que a CucoCloud tem um modelo de sistema muito similar aos objetivos do projeto, em relação às outras aplicações estudadas. Serão retiradas ideias conceptuais da aplicação CucoCloud, que serão aplicadas ao longo do desenvolvimento do sistema deste projeto.

3. PLANEAMENTO DO PROJETO

O desenvolvimento de um projeto exige pelo menos a especificação das seguintes etapas: planeamento e análise de requisitos, desenvolvimento, teste e instalação. Estas etapas são definidas e especificadas tendo em vista uma metodologia de trabalho, que permita a concretização de um projeto de qualidade.

3.1. Metodologia

A metodologia ágil é uma abordagem de desenvolvimento de *software*, em que os requerimentos e soluções envolvem através de esforços colaborativos dos clientes e/ou utilizadores finais e de equipas organizadas, em que cada pessoa tem uma especialidade diferente mas trabalham para um objetivo comum [10].

O desenvolvimento ágil de *software* é dividido em pequenos incrementos que minimizam a quantidade de planeamento e *design*, chamados de iterações. Iterações têm espaços de tempo pequenos, entre uma a quatro semanas. Cada iteração envolve planeamento, análise, *design*, codificação, testes e testes de aceitação. Ao final de cada iteração, o produto é mostrado aos clientes, da qual pode resultar na conclusão do produto ou uma nova iteração, resultante da crítica e análise do cliente. Com esta metodologia, o produto consegue adaptar facilmente a mudanças ou novas funcionalidades [11].

Dentro dos métodos de desenvolvimento ágil, existe uma ampla variedade de iterações do desenvolvimento de *software*. Apesar de existirem práticas comuns entre todos os métodos, cada método focaliza-se em partes diferentes do ciclo de cada iteração [12].

De acordo com estes pressupostos, e porque se considerou que esta seria a metodologia ideal para o desenvolvimentos do projeto, WorkPresence, e considerando que o projeto é apenas constituído por dois membros: o orientador (que pode ser considerado como cliente na perspetiva da metodologia ágil) e o orientando/aluno (programador), estabeleceu-se que as reuniões entre o orientando e o orientador sejam frequentes, definindo pequenos Sprints em cada fase e iterações de desenvolvimento curtas: XP (eXtreme Programming/Programação Extrema).

3.1.1. Extreme Programming

XP é um método desenhado para melhorar a qualidade de *software* e a capacidade de resposta à mudança de requerimentos de *software* pela parte do cliente. Implementa iterações curtas que melhoram a produtividade e reuniões com o cliente, onde novas funcionalidades podem ser introduzidas. Outras características do XP incluem:

- Revisão do código constante.
- O código é testado por todos, quer programadores, quer utilizadores.
- Usar o *design* funcional mais simples.
- Definir e redefinir a arquitetura do projeto constantemente.
- Integrar e testar o código várias vezes ao dia.

Para além destas características, o método XP contém valores que devem ser considerados durante todo o processo de desenvolvimento de *software*. Estes valores incluem: comunicação, simplicidade, feedback, coragem e respeito [13].

3.1.1.1. Comunicação

Em metodologias de desenvolvimento de *software* formais, os requerimentos do sistema são transmitidos aos *developers* através de documentação. No entanto, a metodologia XP favorece uma comunicação verbal, frequente entre todos os seus utilizadores e programadores, de modo a fornecer uma visão partilhada única entre todos os intervenientes [14]. A comunicação entre orientador e orientado foi realizada de acordo com estes objetivos.

3.1.1.2. Simplicidade

XP encoraja começar sempre com a solução mais simples e adicionar mais funcionalidades em iterações seguintes. Assim, o XP difere de outros métodos mais convencionais, focalizando-se nas necessidades do presente em vez de pensar em necessidades futuras. Esta metodologia parte do princípio que os requerimentos do *software* vão ser alterados inevitavelmente. Portanto, se planearmos para um futuro incerto, criamos um risco de programar funcionalidades que terão de ser alteradas ou que nunca vão ser usadas, gastando tempo e recursos [15]. De acordo com estes pressupostos, planeámos e desenvolvemos o projeto WorkPresence.

3.1.1.3. Feedback

Existem várias vertentes de feedback no método XP:

1. Feedback do sistema: Ao testar o código periodicamente, os programadores têm feedback direto do estado do sistema.
2. Feedback do cliente: Após implementar mudanças, a equipa recebe feedback do cliente. Este feedback guia o desenvolvimento do *software* na direção correta.
3. Feedback da equipa: Ao receber feedback do cliente, a equipa irá planear o melhor curso de trabalho que tem de tomar. [14]

O Feedback está diretamente relacionado com comunicação e simplicidade de processo, o que permite que falhas do sistema são facilmente comunicadas entre a equipa e resolvidas rapidamente, devido à simplicidade do código [16]. O feedback do orientador para o orientando foi efetuado em cada reunião de forma comunicativa e simples de modo a esclarecer ou colocar novas questões para cada etapa definida no projeto.

3.1.1.4. Coragem

Os programadores necessitam de coragem para transformar, refazer, o seu código, sempre que necessário. Isto significa rever o código, modificá-lo de modo a tornar futuras iterações mais simples ou até eliminar código desnecessário ou obsoleto. Além disso, coragem também significa persistência. Para resolvermos problemas complicados que nos possam aparecer, necessitamos de persistência até encontrarmos uma solução [14]. O desenvolvimento do projeto foi realizado com esta característica sempre presente.

3.1.1.5. Respeito

O valor de respeito inclui o respeito pela equipa e pelo próprio. Os programadores não devem implementar mudanças que falhem testes previamente concebidos ou que façam atrasar o trabalho dos seus colegas. Membros da equipa respeitam o seu próprio trabalho ao trabalhar por alcançar um padrão de *software* de alta-qualidade [14]. O respeito entre os membros da equipa foi a característica base no desenvolvimento do projeto.

Ao adotar os valores previamente apresentados, respeitamos e valorizamos os pressupostos éticos definidos, no âmbito, de um engenheiro informático. Ninguém se deve sentir inútil ou ignorado. Assim, asseguramos uma produção com uma motivação alta e encorajamos lealdade para a equipa e para o projeto. O valor de respeito tem em vista o trabalho de equipa e depende da implementação dos outros valores [14].

3.2. Etapas do projeto

O planeamento de um projeto tem como objetivo, como o nome indica, estabelecer tarefas/etapas e duração das mesmas, com o objetivo de perceber a viabilidade do projeto, isto é, se o mesmo é possível de concretizar no tempo estipulado e com os recursos definidos.

Neste sentido, estabelecemos um plano da sequência de atividades e o seu tempo previsto durante o processo de desenvolvimento do projeto. O mapa de Gantt que se apresenta, Figura 7, descreve as principais tarefas planeadas.

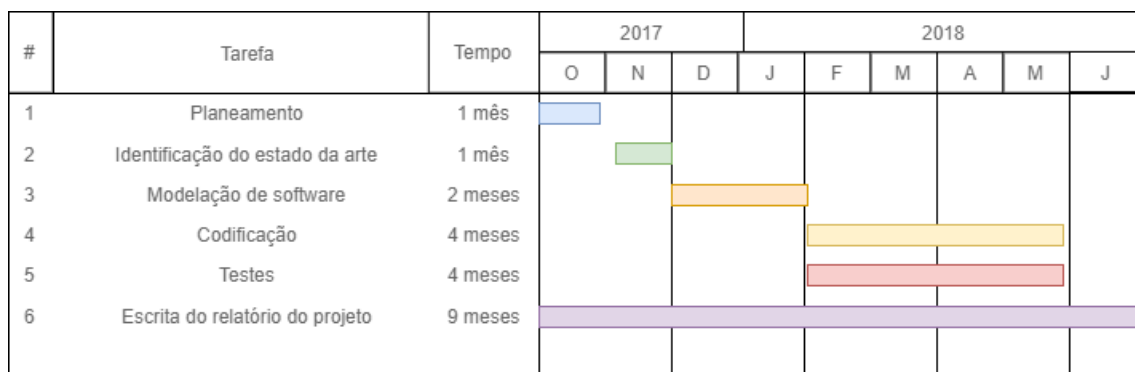


Figura 7 - Mapa de Gantt previsto

De acordo com a Figura 7, podemos verificar que a tarefa mais extensa é a da codificação, realizada em simultâneo com a etapa dos testes. Esta fase de testes é de extrema importância para garantir que o produto final tenha qualidade. Em conjunto com a metodologia de desenvolvimento de *software* e os conceitos de qualidade, os testes contribuem para otimizar o sucesso de um projeto. [17]

É de notar que a codificação é paralela aos testes. Devido à metodologia XP utilizada, o *software* é testado à medida que o código é implementado pelo programador e no final de cada iteração validado pelo cliente.

3.3. Análise de requisitos

Tendo em vista o objetivo específico deste projeto, desenvolvimento de uma solução de registo e controlo de presenças, definiram-se um conjunto de requisitos de sistema que passamos a descrever.

A apresentação dos requisitos foi dividida em requisitos funcionais e requisitos não-funcionais, tendo em consideração os atores do respetivo sistema (utilizadores e administradores). É de notar que os administradores também têm acesso a todos os requisitos definidos para os utilizadores (ex. funcionários).

3.3.1. Requisitos funcionais

O sistema deve permitir o registo de diferentes utilizadores: funcionários da organização e administradores de sistema. Cada um destes utilizadores deverá ter privilégios previamente estabelecidos.

Os administradores terão acesso a funcionalidades que permitirão controlar a sua organização e dados dos seus funcionários. Decidiu-se, por motivos de integridade, que só poderá existir uma conta de administrador por organização. Se o papel de administrador precisar de ser partilhado por várias pessoas, estes terão de usar as mesmas credenciais.

Os funcionários terão funcionalidades dedicadas e privilégios que lhes permitem visualizar os seus dados de perfil e dados de registo de presenças (entrada e saída).

Estas funcionalidades, requisitos funcionais do sistema, encontram-se listadas nos seguintes sub-pontos:

3.3.1.1. Administradores

- Gerir funcionários;
- Gerir departamentos;
- Gerir horários;
- Registo de check-in/out manual dos seus funcionários;
- Analisar as presenças dos funcionários;
- Ver a localização das presenças dos funcionários;

- Analisar estatísticas de presenças dos funcionários.

3.3.1.2. Funcionários

- Registrar o Check-in/out na organização;
- Visualizar o seu horário;
- Analisar o seu histórico de presenças;
- Gerir os seus dados/perfil.

3.3.2. Requisitos não-funcionais

Tendo em vista os objetivos do projeto, definiram-se os seguintes requisitos não funcionais:

1. Intuitivo: a interface do sistema deve ser fácil de compreender, de modo a que os utilizadores não precisem de muito tempo a aprender a utilizar o sistema na sua totalidade.
2. Interface Web: os utilizadores terão acesso, a partir de qualquer plataforma web, às funcionalidades do sistema.
3. Interface móvel: o sistema terá uma aplicação móvel, na qual estarão acessíveis as funcionalidades dos funcionários.
4. Sincronização Cloud: os dados de cada utilizador podem ser acedidos ou modificados a partir de qualquer dispositivo.
5. Desempenho: é preciso testar a aplicação em termos de performance, durante o seu desenvolvimento. Especificamente, o tempo de resposta do servidor para a aplicação web e para a aplicação móvel.

Após o levantamento dos requisitos, do sistema Workpresence procedemos à modelação do sistema, utilizando para tal a linguagem de modelação *Unified Modelling Language*, que nos permitirá uma compreensão mais ágil relativa às funcionalidades requeridas pelo sistema.

3.4. Modelação UML

A linguagem *Unified Modelling Language* (UML) é uma linguagem de modelação que tenta padronizar uma maneira de visualizar um sistema de *software* [18]. Esta linguagem, UML, visa melhorar a qualidade de análise de sistemas e *design*, através de práticas que foram padronizadas ao longo do tempo. Ao utilizar UML na análise e *design* do sistema, é possível obter uma compreensão melhor entre a equipa de desenvolvimento e os clientes quanto aos requerimentos do sistema e os processos necessários para chegar a esses requerimentos [19].

Nesta secção, serão apresentados alguns diagramas representativos das funcionalidades do projeto, criados durante a análise do sistema para uma melhor compreensão das funcionalidades que o sistema deve conter de modo a poder cumprir com os objetivos.

3.4.1. Diagrama de contexto

Ainda que o diagrama de contexto não corresponde à modelação UML, apresentamos o mesmo uma vez que consideramos que a sua modelação permite obter uma visão geral básica de todo o sistema ou processo a ser analisado ou modelado.

O diagrama de contexto, na engenharia de *software*, é um diagrama que define a fronteira entre o sistema e o seu ambiente, mostrando as entidades que interagem com ele [20]. Neste caso, as entidades que irão interagir com o nosso sistema são os administradores e os funcionários da organização.

O administrador é o responsável por gerir os funcionários de uma organização (que registou previamente), através da aplicação web. Os privilégios dos administradores permitem-lhes criar e gerir funcionários, horários, presenças e departamentos numa organização. Estes têm acesso a todos os registos de dados da aplicação e podem também adicionar manualmente, um registo de entrada ou saída de funcionário, no caso de se registar alguma falha de sistema.

Os administradores do sistema têm também privilégios de acesso a estatísticas dos funcionários (ex: número de horas de trabalho, dias atrasados, etc..) e ainda acesso a um mapa contendo a última localização registada de cada funcionário.

As ações dos funcionários, no sistema, focalizam-se em particular nos registos de presença (entrada e saída). Cada funcionário irá utilizar a aplicação, instalada no seu *smartphone*, para registar o seu check-in e check-out. Adicionalmente, terão acesso aos seus horários, geridos pelo administrador, e aos dados estatísticos relativos a horas de trabalho, horas extras, presenças em geral e departamento a que pertencem.

Estas funcionalidades, requisitos, são apresentados graficamente, para cada entidade (funcionários e administradores), no diagrama de contexto da Figura 8 e Figura 9:

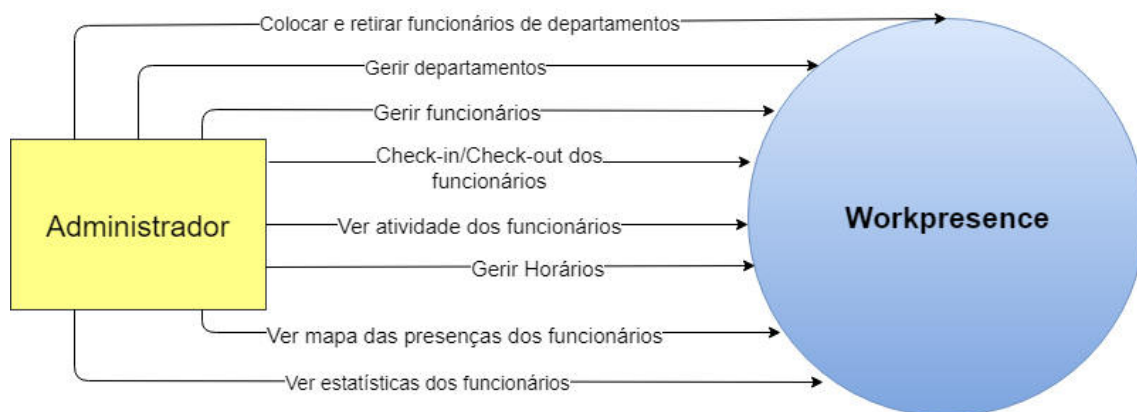


Figura 8 - Diagrama de contexto do administrador

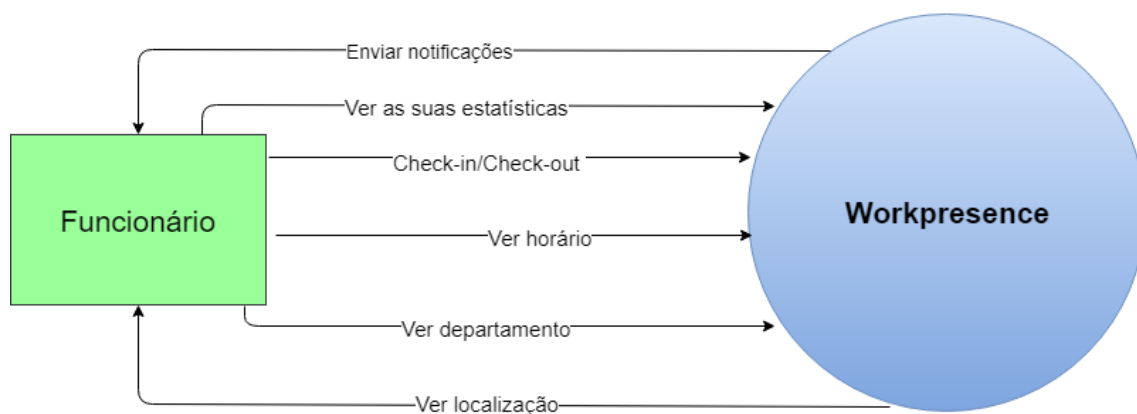


Figura 9 - Diagrama de contexto do funcionário

3.4.2. Diagrama de casos de uso

Os diagramas de caso de uso descrevem um conjunto de ações (casos de uso) que podem ser executadas, dentro de um ou mais sistema, por utilizadores externos do sistema (atores). Cada caso de uso deve fornecer um resultado observável e valioso aos atores ou outros intervenientes do sistema [21].

Um caso de uso é uma descrição escrita de como os utilizadores efetuam tarefas dentro de um sistema. Representa como o sistema responde e se comporta a um pedido, do ponto de vista do utilizador. Cada caso de uso é representado através de uma sequência de passos, começando com o objetivo do utilizador e acabando quando esse objetivo é cumprido [22].

A Figura 10, representa o diagrama de caso de uso para cada ator do sistema (Funcionário, Administrador).

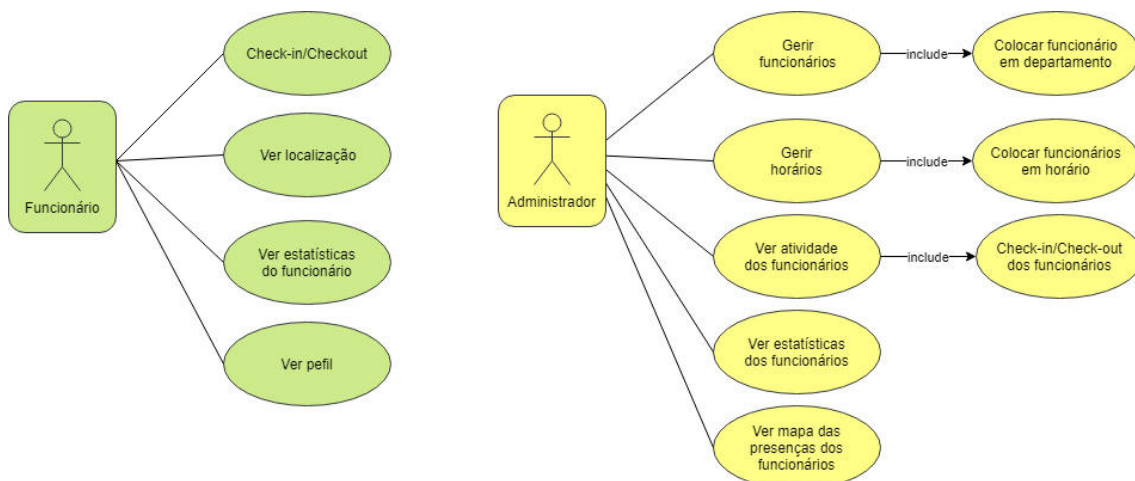


Figura 10 - Diagrama de Casos de Uso do Workpresence

3.4.3. Descrições de caso de uso/Diagramas de sequência

Cada descrição de caso de uso terá associado um diagrama de sequência. Os diagramas de sequência mostram como os objetos de um caso de uso específico, interagem entre si. Um diagrama de sequência é composto por atores e objetos que são representados através de linhas verticais. As interações são mostradas através de setas que descrevem a ação entre os objetos e a direção de interação. As ações estão ordenadas verticalmente pela linha de tempo [22].

Apresentam-se, no subcapítulo seguinte, os diagramas de sequência para o Ator Funcionários, de acordo com os Casos de usos especificados anteriormente. Para resumir esta secção, algumas tabelas menos prioritárias estão no ANEXO D1.

3.4.3.1. Descrição casos de uso - Funcionários

3.4.3.1.1. Check-in

O caso de uso “Check-in” servirá para registar a hora inicial da presença do funcionário, iniciando uma sessão do funcionário. Denominou-se sessão ao intervalo de tempo entre um “Check-in” de um funcionário e ao “Checkout” sucessivo.

Nome	Check-in
Objetivo	Marcar a presença do utilizador
Atores	Funcionário
Pré-condição	Login efetuado e sem Check-in efetuado
Prioridade	Alta
Fluxo Principal	<ol style="list-style-type: none"> 1. O funcionário clica no botão “Check-in” na interface principal. 2. O sistema verifica a localização do dispositivo. 3. O sistema inicia uma sessão e regista a hora, data, utilizador e localização atual. 4. O sistema apresenta mensagem de confirmação.
Fluxo Secundários	<ol style="list-style-type: none"> 2. A) O sistema verifica que o dispositivo não tem geolocalização ativa e termina a operação com uma mensagem de erro.
Exceções	<ol style="list-style-type: none"> 3. A) Não existe ligação à base de dados e o sistema cancela a operação.
Pós-condição	O botão “Check-in” altera-se para o botão “Checkout”.
Casos de teste	<ol style="list-style-type: none"> 1. Verificar se o sistema deteta a geolocalização efetivamente.

Tabela 2 - Descrição de caso de uso "Check-in"

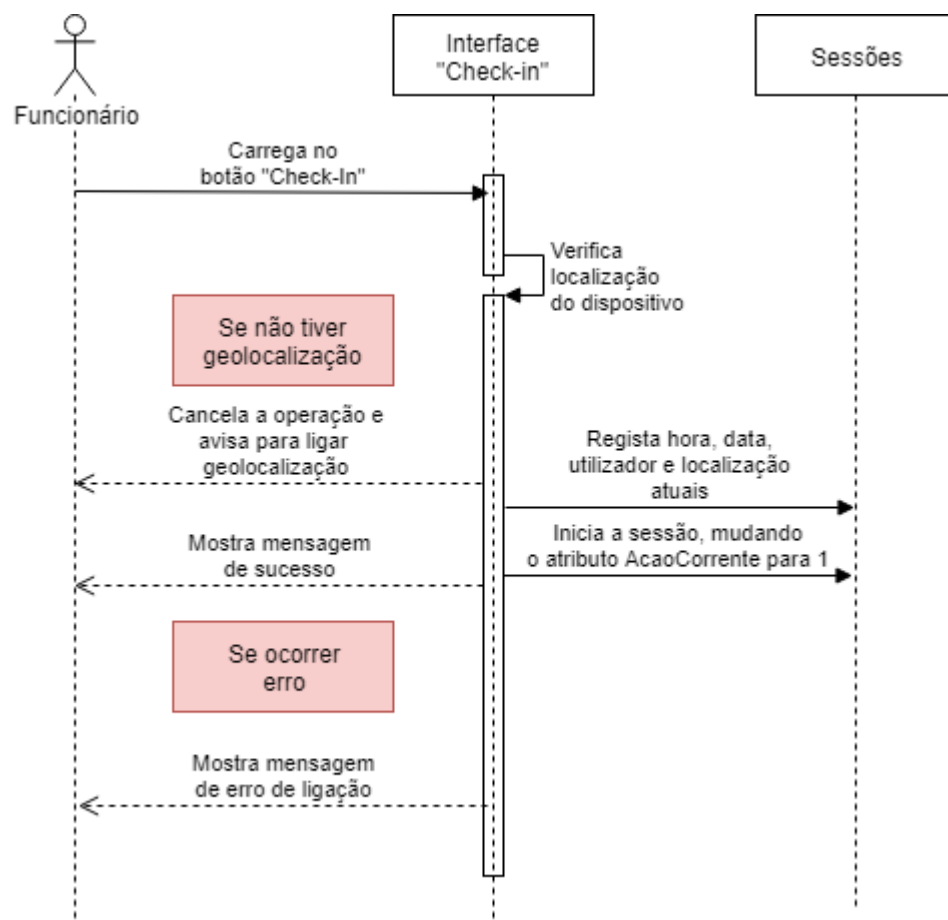
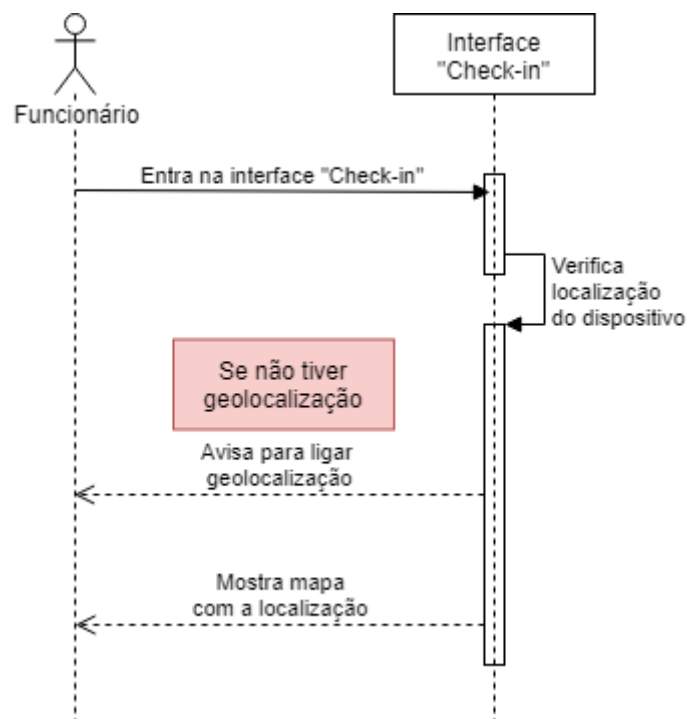


Figura 11 - Diagrama de sequência do "Check-in"

3.4.3.1.2. Ver localização

Nome	Ver Localização
Objetivo	Mostrar a localização atual do utilizador
Atores	Funcionário
Pré-condição	Login efetuado
Prioridade	Baixa
Fluxo Principal	<ol style="list-style-type: none"> 1. O funcionário acede à interface principal da aplicação móvel. 2. O sistema verifica a localização do dispositivo. 3. O sistema mostra um mapa com a localização atual do utilizador.
Fluxo Secundários	<ol style="list-style-type: none"> 2. A) O sistema verifica que o dispositivo não tem geolocalização ativa e mostra uma mensagem com o erro.
Exceções	
Pós-condição	
Casos de teste	<ol style="list-style-type: none"> 1. Verificar se o sistema mostra as coordenadas corretas no mapa.

Tabela 3 - Descrição de caso de uso "Ver Localização"**Figura 12 - Diagrama de sequência "Ver localização"**

3.4.3.1.3. Ver estatísticas do funcionário

Nome	Ver estatísticas do funcionário
Objetivo	Mostrar as estatísticas do funcionário
Atores	Funcionário
Pré-condição	Login efetuado
Prioridade	Média
Fluxo Principal	<ol style="list-style-type: none"> 1. O funcionário acede à interface “Estatísticas”. 2. O sistema mostra a atividade mais recente. 3. A seguir, o funcionário pode escolher duas datas. <ol style="list-style-type: none"> a. O sistema mostra só os dados estatísticos associados às datas escolhidas.
Fluxo Secundários	<ol style="list-style-type: none"> 2. A) Ocorre um erro e o sistema mostra uma mensagem do erro ocorrido.
Exceções	
Pós-condição	
Casos de teste	<ol style="list-style-type: none"> 1. Verificar se o filtro por datas funciona corretamente.

Tabela 4 - Descrição de caso de uso "Ver estatísticas de funcionário"

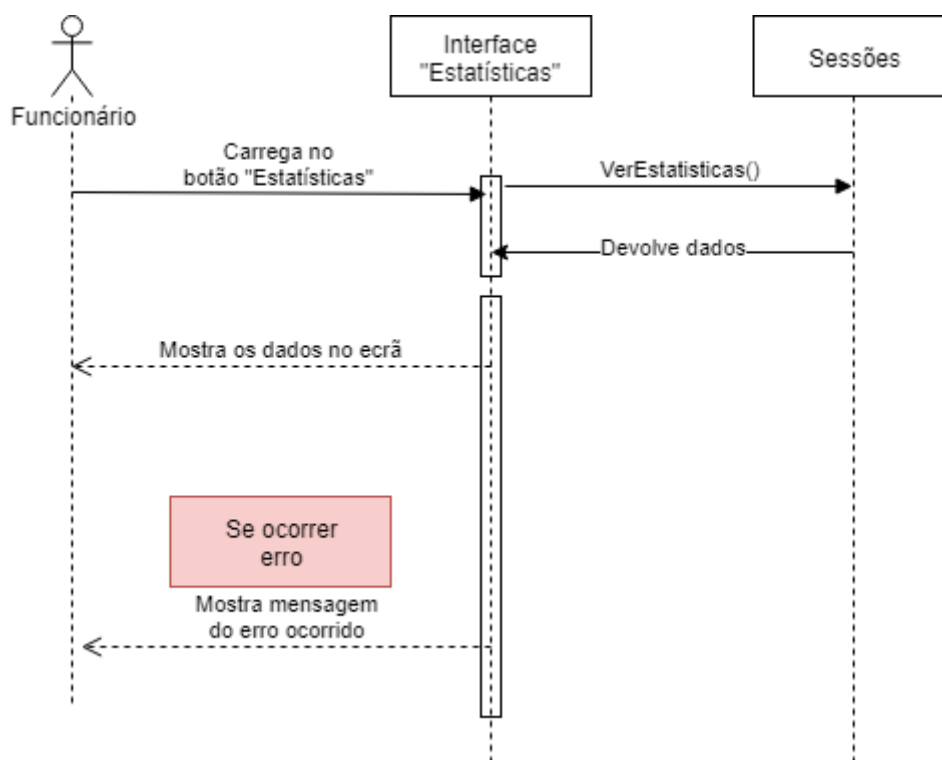


Figura 13 - Diagrama de Sequência "Ver estatísticas do funcionário"

3.4.3.1. Descrição casos de uso – Administrador

3.4.3.1.1. Gerir funcionários

Nome	Gerir funcionários
Objetivo	Visualizar, inserir, editar e apagar funcionários
Atores	Administrador
Pré-condição	Login de administrador efetuado
Prioridade	Média
Fluxo Principal	<ol style="list-style-type: none"> 1. O administrador acede à interface “Funcionários”. 2. O sistema mostra os dados dos funcionários da organização do administrador e os botões que abrem os menus das outras funcionalidades (inserir, editar e apagar.). 3. O administrador clica no botão cuja funcionalidade pretende efetuar. 4. O sistema apresenta um menu adicional com os campos a preencher, relativos à funcionalidade escolhida no ponto 3. 5. O administrador preenche os campos e clica no botão para concluir a operação. 6. O sistema processa os dados recebidos e mostra uma mensagem de sucesso.
Fluxo Secundários	<ol style="list-style-type: none"> 4. A) O administrador pode fechar ou abrir outro menu de outra funcionalidade a qualquer momento. 6. A) Se ocorrer um erro, o sistema mostra uma mensagem do erro que ocorreu.
Exceções	
Pós-condição	
Casos de teste	

Tabela 5 - Descrição de caso de uso "Gerir Funcionários"

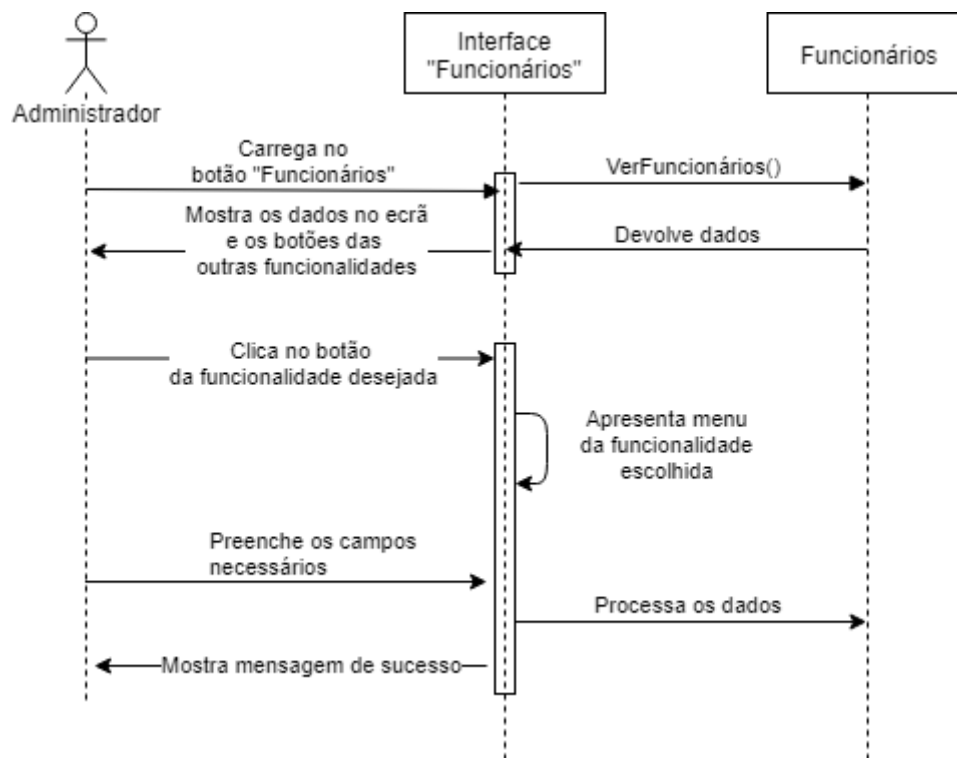


Figura 14 - Diagrama de sequência "Gerir Funcionários"

3.4.3.1.2. Ver atividade dos funcionários

Nome	Ver atividade dos funcionários
Objetivo	Mostrar a atividade dos funcionários da organização ao administrador
Atores	Administrador
Pré-condição	Login de administrador efetuado
Prioridade	Alta
Fluxo Principal	<ol style="list-style-type: none"> 1. O administrador entra na interface “Atividade dos funcionários”. 2. O sistema mostra a atividade mais recente dos funcionários da organização. 3. O administrador pode restringir a atividade mostrada através dos vários filtros disponíveis (por nome, por data, etc...)
Fluxo Secundários	
Exceções	
Pós-condição	
Casos de teste	

Tabela 6 - Descrição de caso de uso "Ver atividade de funcionário"

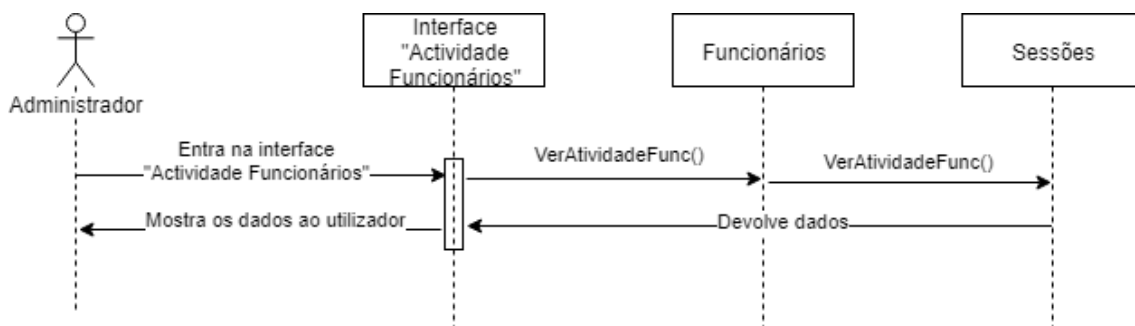


Figura 15 - Diagrama de sequência "Ver Atividade de Funcionário"

3.4.4. Modelo ER e semântica de dados

Em engenharia de *software*, o modelo ER é um modelo de data abstrato, que define informação ou estrutura de informação que pode implementado numa base de dados relacional. Neste ponto, apresentamos o modelo ER e o respetivo dicionário de dados, que permite descrever as entidades e atributos do modelo.

3.4.4.1. Modelo ER

O esquema seguinte define o modelo ER deste projeto (Figura 16):

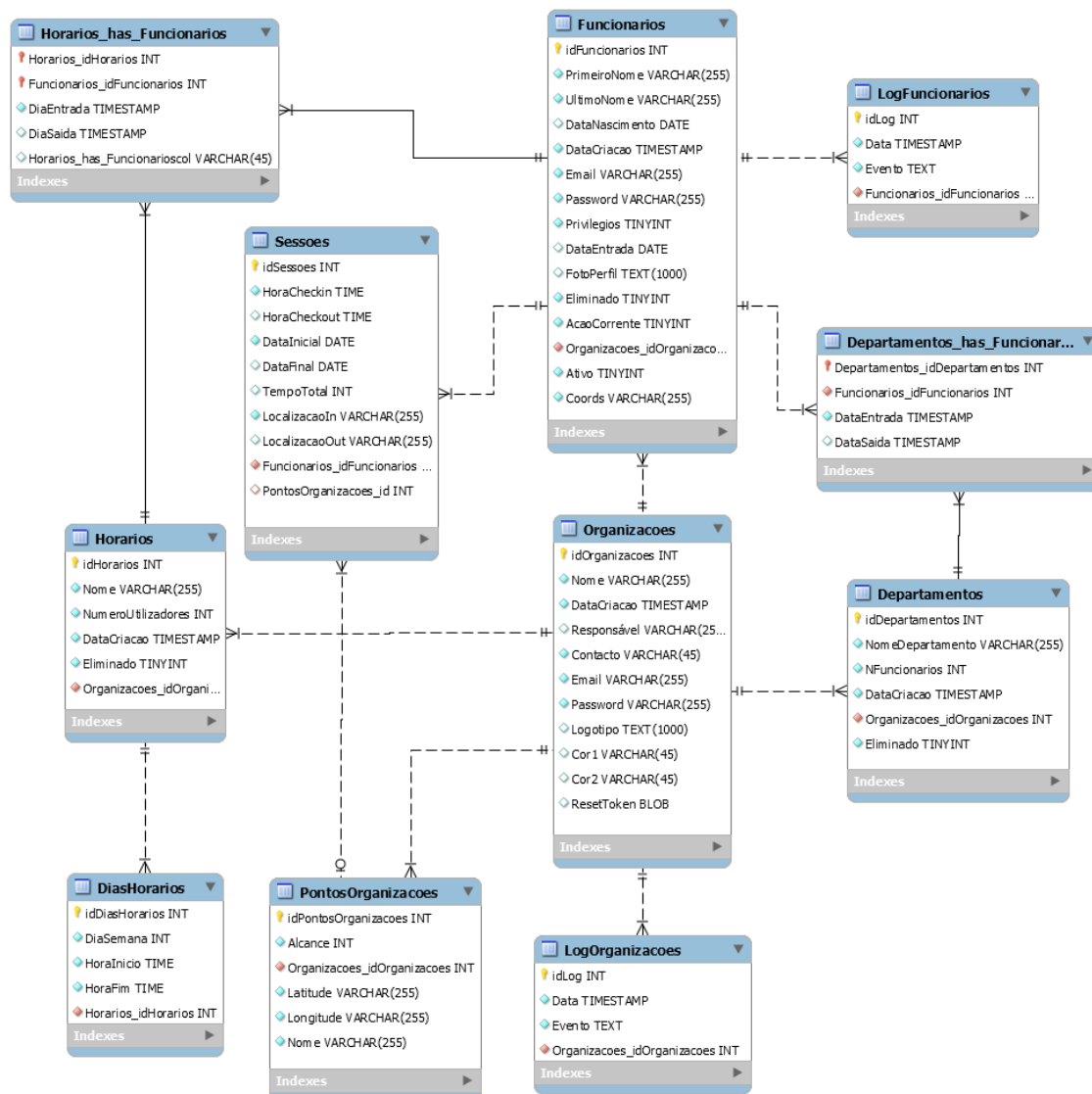


Figura 16 - Modelo ER

3.4.4.2. Semântica de dados

Nesta secção é apresentado o dicionário de dados, que descreve os campos de cada classe, os seus formatos e as suas restrições. Adicionalmente, também são apresentadas as descrições dos métodos de cada classe.

3.4.4.2.1. Organizações

Nome	Tipo de dados	Tamanho	Descrição	Restrições
idOrganizações	INT	11	Chave primária da tabela	Maior que 0. Não nulo.
Nome	VARCHAR	255	Nome da organização	Não nulo.
DataCriação	TIMESTAMP	8	Data e hora da criação da organização no sistema	Não nulo Omissão: data do sistema
Responsável	VARCHAR	255	Nome do responsável pela organização	Não nulo
Contacto	VARCHAR	45	Número de telemóvel do responsável	Não nulo
Email	VARCHAR	255	Email da organização e credencial para entrar na aplicação	Não nulo
Password	VARCHAR	255	Credencial para entrar na aplicação como administrador da organização	Não nulo
Logótipo	VARCHAR	255	Caminho para a imagem do logótipo da organização	Nulo
Cor1	VARCHAR	45	Código hexadecimal da primeira cor da organização	Nulo
Cor2	VARCHAR	45	Código hexadecimal da segunda cor da organização	Nulo

Tabela 7 - Dicionário de dados da tabela "Organizações"

Nome da função	Descrição
LoginOrganização	Verifica as credenciais de uma organização.

Tabela 8 - Operações da tabela "Organizações"

3.4.4.2.2. Funcionários

Nome	Tipo de dados	Tamanho	Descrição	Restrições
idFuncionarios	INT	11	Chave primária da tabela	Maior que 0. Não nulo.
PrimeiroNome	VARCHAR	255	Primeiro nome do funcionário	Não nulo.
UltimoNome	VARCHAR	255	Último nome do funcionário	Não nulo.
DataNascimento	DATE	12	Data de nascimento do funcionário	Nulo
DataCriação	TIMESTAMP	8	Data e hora da criação do funcionário no sistema	Não nulo Omissão: data do sistema
Email	VARCHAR	255	Email do funcionário e credencial para entrar na aplicação	Não nulo
Password	VARCHAR	255	Credencial para entrar na aplicação como funcionário da organização	Não nulo
Privilégios	TINYINT	1	Nível de privilégios do funcionário.	Não Nulo Omissão:0
FotoPerfil	VARCHAR	255	Caminho para a imagem de foto de perfil	Nulo
Eliminado	TINYINT	1	Se o valor for 1, indica que o funcionário foi eliminado	Não Nulo Omissão:0
AcaoCorrente	TINYINT	1	Indica se a próxima ação do funcionário será um check-in (0) ou um check-out (1)	Não Nulo Omissão:0
idOrganizacoes	INT	11	Chave estrangeira para as organizações	Não Nulo
Ativo	TINYINT	1	Indica se o funcionário está a desempenhar funções na organização	Não Nulo Omissão:1
Coords	VARCHAR	255	Indica as últimas coordenadas do funcionário.	Não Nulo Omissão:'''

Tabela 9 - Dicionário de dados da tabela "Funcionários"

Nome da função	Descrição
FiltraFuncionários	Função de pesquisa da tabela funcionários. Devolve os funcionários consoante os filtros estabelecidos pelo administrador.
LoginFuncionário	Verifica as credenciais do utilizador.
MostraFuncionário	Devolve os dados de um funcionário específico.
VerLocalização	Mostra as coordenadas geográficas da última atividade efetuada pelos funcionários.

Tabela 10 - Operações da tabela "Funcionários"

3.4.4.2.3. Sessões

Nome	Tipo de dados	Tamanho	Descrição	Restrições
idSessões	INT	11	Chave primária da tabela	Maior que 0. Não nulo.
HoraCheckIn	TIME	12	Hora do check-in	Não nulo.
HoraCheckOut	TIME	12	Hora do check-out	Nulo.
DataCheckIn	DATE	12	Data do check-in	Não Nulo
DataCheckOut	DATE	12	Data do check-out	Nulo
TempoTotal	INT	11	Tempo total que durou a sessão.	Nulo
LocalizaçãoIn	VARCHAR	255	Localização onde foi feito o check-in	Não nulo
LocalizaçãoOut	VARCHAR	255	Localização onde foi feito o check-out	Nulo
idFuncionarios	INT	11	Chave estrangeira para o funcionário	Maior que 0. Não nulo.
idPontos Organizacao	INT	11	Chave estrangeira para o ponto da organização onde fez o check-in	Maior que 0. Não nulo.

Tabela 11 - Dicionário de dados da tabela "Sessões"

Nome da função	Descrição
FuncionárioCheckIn	Realiza o check-in de um funcionário.
FuncionárioCheckOut	Realiza o check-out de um funcionário.
VerAtividade	Devolve as presenças efetuadas, recentemente, de uma organização.

Tabela 12 - Operações da tabela "Sessões"

3.4.4.2.4. Departamentos

Nome	Tipo de dados	Tamanho	Descrição	Restrições
idDepartamentos	INT	11	Chave primária da tabela	Maior que 0. Não nulo.
NomeDepartamento	VARCHAR	255	Nome do departamento	Não nulo.
NFuncionarios	INT	11	Número de funcionários no departamento	Não Nulo Omissão:0
DataCriacao	TIMESTAMP	8	Data e hora da criação do departamento no sistema	Não nulo Omissão: data do sistema
idOrganizacoes	INT	11	Chave estrangeira para as organizações	Não Nulo
Eliminado	TINYINT	1	Se o valor for 1, indica que o departamento foi eliminado	Não Nulo Omissão:0

Tabela 13 - Dicionário de dados da tabela "Departamentos"

3.4.4.2.5. Horários

Nome	Tipo de dados	Tamanho	Descrição	Restrições
idHorários	INT	11	Chave primária da tabela	Maior que 0. Não nulo.
Nome	VARCHAR	255	Nome do horário	Não nulo.
NumeroUtilizadores	INT	11	Número de funcionários no horário	Não Nulo Omissão:0
DataCriacao	TIMESTAMP	8	Data e hora da criação do horário no sistema	Não nulo Omissão: data do sistema
idOrganizacoes	INT	11	Chave estrangeira para as organizações	Não Nulo
Eliminado	TINYINT	1	Se o valor for 1, indica que o horário foi eliminado	Não Nulo Omissão:0

Tabela 14- Dicionário de dados da tabela "Horários"

Nome da função	Descrição
InserirFuncHorário	Atribui um horário a um funcionário

Tabela 15 - Operações da tabela "Horários"

3.4.4.2.6. PontosOrganizações

Nome	Tipo de dados	Tamanho	Descrição	Restrições
idPontosOrganizações	INT	11	Chave primária da tabela	Maior que 0. Não nulo.
Nome	VARCHAR	255	Nome do ponto geográfico	Não nulo.
Latitude	VARCHAR	255	Latitude do ponto geográfico	Não nulo.
Longitude	VARCHAR	255	Longitude do ponto geográfico	Não nulo.
idOrganizacoes	INT	11	Chave estrangeira para as organizações	Não Nulo
Alcance	INT	11	Tamanho do raio do ponto de entrada em metros	Não Nulo Omissão:50

Figura 17 - Operações da tabela "PontosOrganizações"

3.4.5. Diagrama de classes

Os diagramas de classes mostram as classes de um sistema, as suas relações (incluindo hierarquia, agregação e associação), operações e atributos de classes. Os diagramas de classes são usados para uma grande variedade de objetivos, incluindo modelação concetual e modelação de *design* detalhado [23].

O diagrama de classes desenvolvido para este projeto apresenta-se na Figura 18:

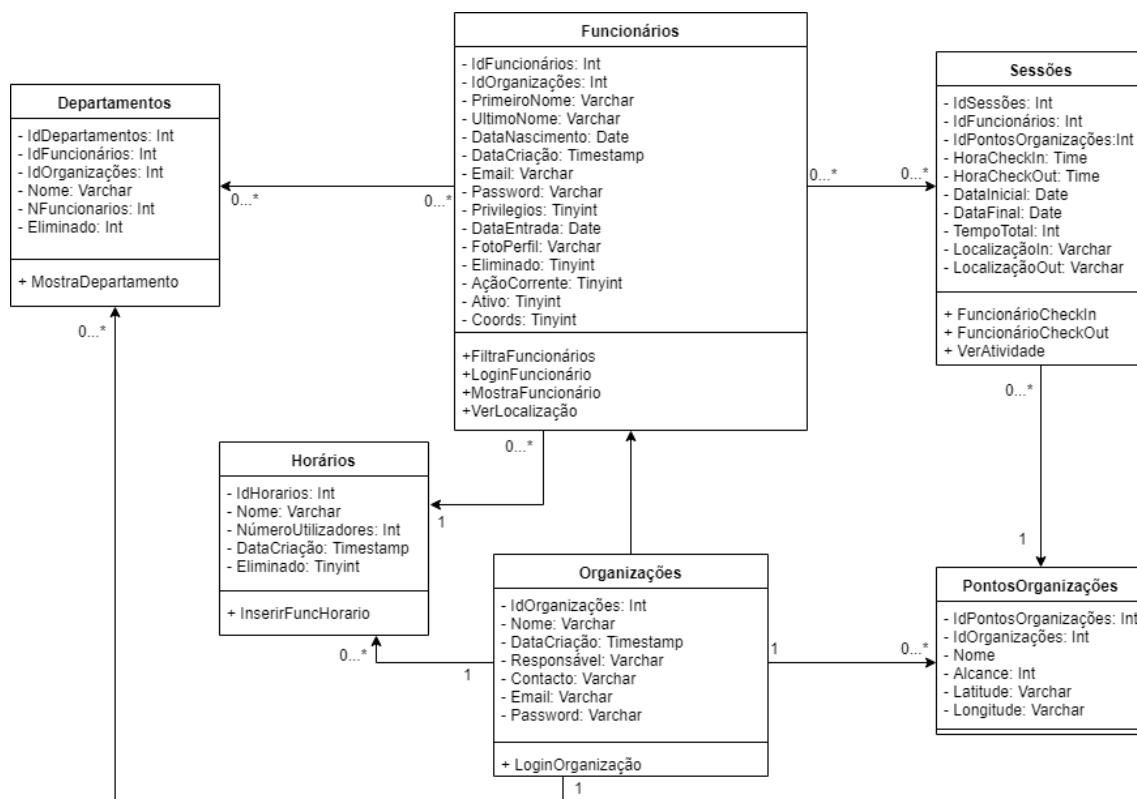


Figura 18 - Diagrama de classes⁴

A aplicação terá o seu foco em três classes principais, definidas como: Funcionários, Organizações e Sessões.

A classe Organizações regista os dados respetivos a cada organização. Nesta classe encontra-se o Email e a Password que vão ser utilizadas como credenciais para os administradores. A sua função principal é permitir distinguir entre as várias organizações que vão utilizar o *software* e distinguir um funcionário com privilégios dito normal de um administrador de sistema.

⁴ É de notar que só foram escritas as funções mais importantes para o funcionamento do sistema. As funções mais básicas (ex: criar, ler, editar e apagar) não estão representadas para abreviar este relatório.

A classe Funcionários contém os dados relativos aos funcionários de cada organização e é a classe mais importante do sistema, porque irá interagir com a maioria das funções do sistema. Esta classe contém os dados relativos a cada funcionário e permite identificar cada funcionário de forma individual. Os campos/atributos Email e Password serão utilizados como credenciais para o funcionário entrar na aplicação.

A classe Sessões é a classe que irá registar os dados de cada check-in e check-out dos funcionários e permitirá posteriormente analisar dados e realizar estatísticas. Ao realizar-se um check-in, por parte de um funcionário, o sistema cria um novo registo dentro da classe. Os campos da classe sessão são:

- IdFuncionários: regista o funcionário que fez o check-in;
- IdPontosOrganizações: regista o ponto de entrada no qual fez check-in;
- HoraCheckIn: regista a hora em que fez check-in;
- DataCheckIn: regista o dia em que fez check-in;
- LocalizaçãoIn: regista o ponto geográfico específico onde fez check-in;

Todos os outros campos, constante na classe, ficam nulos até o funcionário proceder ao check-out. Ao fazer check-out, os restantes campos são preenchidos identicamente terminando a sessão. O campo TempoTotal é registado de acordo com o tempo que durou a sessão.

Este método de sessões foi criado de forma a otimizar o sistema, e de modo a que este possa calcular as estatísticas relativas aos distintos funcionários. Em vez de termos duas classes independentes para os check-ins e check-outs e de termos de verificar a que check-in tem de corresponder cada check-out, temos uma única classe na qual essa ligação já está formada, tornando muito mais simples e eficaz o processo que permite a análise de dados e o cálculo de estatísticas.

4. IMPLEMENTAÇÃO

Neste capítulo, apresentam-se as principais tecnologias e ferramentas utilizadas no projeto, bem como o processo de desenvolvimento de algumas das funcionalidades mais importantes do projeto. São também descritos os desafios de maior complexidade encontrados ao longo do projeto e as respectivas soluções definidas.

4.1. Tecnologias utilizadas no projeto

Nesta subsecção são listadas e descritas, de uma forma breve, as tecnologias e ferramentas mais importantes que foram utilizadas no desenvolvimento do projeto.

4.1.1. HTML

HTML é uma linguagem de marcação utilizada para produzir páginas na Web. Documentos HTML são interpretados por navegadores Web de modo a construir interfaces e conteúdo para o utilizador [24].

O HTML pode ser considerado o bloco de construção mais básico de uma página Web e é utilizado para criar e visualmente representar uma página web. Ele determina os conteúdos de uma página Web mas não a sua funcionalidade, ao contrário de linguagens como PHP e JavaScript que focam na interatividade do site com o utilizador em vez do seu conteúdo. Esta linguagem tornou-se no padrão e base para criação de qualquer página Web. Como uma grande componente do projeto é uma aplicação Web, esta linguagem é de grande importância e foi fundamental no desenvolvimento do sistema WorkPresence.

4.1.2. CSS e Bootstrap

Cascading Style Sheets (CSS) é uma linguagem de folhas de estilo utilizada para definir a apresentação de documentos escritos numa linguagem de marcação, como HTML ou XML. O seu principal benefício é fornecer a separação entre o formato e o conteúdo de um documento [25].

O CSS permite criar e alterar um conjunto de propriedades de estilo das páginas Web. Estas propriedades alteram o *design* e a visualização de qualquer elemento HTML, seguindo um conjunto de regras impostas pelo programador. Assim, com esta linguagem, conseguem-se criar *layouts* e *designs* específicos de modo eficiente.

Especificamente, no projeto, vai ser utilizado uma *framework* de CSS e jQuery chamado Bootstrap. Esta *framework* está pré-configurado para alterar os elementos básicos de HTML não só para interfaces mais complexos, mas também para tornar páginas web responsivas, de modo a modificar a interface da página de acordo com o ecrã do dispositivo que está a ser utilizado pelo utilizador. Assim, a página consegue ser visualizada de modo intuitivo e eficiente quer nos pequenos ecrãs dos dispositivos móveis quer nos maiores ecrãs dos computadores, tornando o sistema responsivo.

4.1.3. Javascript, AJAX e jQuery

A linguagem Javascript é considerada uma das três linguagens essenciais para o desenvolvimento Web (sendo as outras duas HTML e CSS). O Javascript é a principal linguagem responsável pelo comportamento das páginas web, sendo capaz de tornar as páginas web dinâmicas, interativas e intuitivas para o utilizador. No entanto, a tecnologia fulcral que o Javascript proporciona é o AJAX.

O AJAX (Asynchronous JavaScript and XML) não é uma ferramenta de programação, mas sim uma técnica, que utiliza as tecnologias Javascript e XML, para processar dados externos assincronamente, sem recarregar a página em que o utilizador se encontra. Deste modo, conseguimos ter uma interação permanente com o servidor sem colocar em causa a dinâmica da aplicação web com o utilizador. O AJAX permite manipular e visualizar registos em tempo real (ex: ver as presenças dos funcionários à medida que fazem check-in).

jQuery é uma biblioteca de Javascript concisa, eficiente e com bastantes funcionalidades. Não só funciona com uma grande variedade de *browsers*, mas simplifica algumas funcionalidades do Javascript como transferência de ficheiros, manipulação de elementos HTML, animação e AJAX [26]. No projeto, a biblioteca jQuery substitui a linguagem nativa Javascript por completo, simplificando a implementação e eficácia da aplicação Web e, consequentemente, tornando a aplicação Web mais versátil a mudanças do sistema.

4.1.4. PHP

O PHP: Hypertext Preprocessor (PHP) é uma linguagem interpretada livre, usada originalmente apenas para o desenvolvimento de aplicações presentes e interativas no lado do servidor, capazes de gerar o conteúdo dinâmico nas páginas Web. Há muitas outras linguagens de programação para a Web, tais como ASP.Net, Java, Python. No entanto, decidiu-se utilizar o PHP pelas seguintes razões:

1. A experiência do autor nesta linguagem de programação. O autor teve várias disciplinas dedicadas à programação na Internet, incluindo Programação em PHP. Adicionalmente, o autor também trabalhou com PHP extensivamente no projeto de licenciatura.
2. A enorme quantidade de informação atualizada que existe sobre a linguagem. Sendo uma linguagem bastante conhecida e utilizada, PHP está constantemente a ser discutida e desenvolvida. Consequentemente, conseguimos encontrar bastante informação em livros, artigos e em documentação oficial em como trabalhar e utilizar a linguagem efetivamente.
3. Adaptação. Devido ao seu desenvolvimento durante muitos anos, PHP tem muitas funções e potencial. Assim, conseguimos adaptar o nosso código a mudanças do sistema quer seja otimizações do código quer seja a necessidade de implementar novas funcionalidades.

No projeto, a linguagem PHP é utilizada maioritariamente na implementação dos serviços Web, que servem para estabelecer comunicação entre o servidor e as duas aplicações Web e mobile. O PHP tem uma grande variedade de funções que permitem estabelecer conexões a bases de dados, executar *queries* e tratar os dados recebidos. Assim, torna-se numa ferramenta ideal para estabelecer interligações dentro de todo o sistema.

4.1.5. MySQL

O MySQL é um sistema de gestão de base de dados (SGBD) que utiliza a linguagem SQL (Structured Query Language) como interface standard de acesso aos dados. É gratuito, possui facilidades para certas operações Web (como a paginação dos resultados das consultas) e é atualmente um dos bancos de dados mais populares nas aplicações Web, com mais de 10 milhões de instalações pelo mundo [27]

Para além do seu grande desempenho e estabilidade, o MySQL tem grande compatibilidade com a linguagem PHP, tendo este um módulo de interface próprio. Este módulo de interface, contendo funções nativas preparadas de raiz para lidar com conexões de base de dados e manipulação de dados, tem-se mantido regularmente atualizado e testado. Isto não só garante um nível de segurança mais alto comparando com outros SGBDs, mas também uma maior eficácia a programar, para além do acesso a uma grande quantidade de documentação disponível.

No projeto, MySQL é utilizado para criar e manipular toda a base de dados. Através da criação de procedimentos e funções MySQL, é possível tornar a base de dados versátil a mudanças do sistema.

4.1.6. Android

O Android é um sistema operativo destinado a dispositivos móveis (smartphones e tablets), desenvolvido pela Google. Lançado em setembro de 2008, o Android recebeu muitas atualizações e lançamentos desde então, até chegar à sua versão corrente: “Pie” (versão 9).

O Android é o sistema operativo mais vendido do mundo em *smartphones* desde 2011 e em *tablets* desde 2013. Em maio de 2017, registaram-se mais de dois mil milhões de utilizadores ativos mensais, sendo o sistema operativo instalado com a maior base de utilizadores. Em dezembro de 2018, a loja virtual Google Play registava mais de 2,6 milhões de apps [28].

Tendo em conta não só os números impressionantes de utilizadores de Android apresentados anteriormente, mas também a experiência e o conhecimento do autor em Android, adquiridos nas unidades curriculares frequentadas no Mestrado, tornou-se evidente e óbvio utilizar este sistema operativo, Android, para o desenvolvimento do projeto.

As aplicações Android podem ser construídas utilizando as linguagens Java, Kotlin ou C++ juntamente com o kit de desenvolvimento de *software* Android (SDK). Para este projeto, a linguagem Java foi escolhida devido também à experiência prévia do autor nesta linguagem. Adicionalmente, vai ser utilizado o IDE Android Studio que contém a integração entre a linguagem Java e o Android SDK, entre outras ferramentas que aceleram o processo de desenvolvimento da aplicação móvel. É de notar que o Android Studio é o IDE oficial da Google e o único recomendado pelo mesmo.

4.1.7. Web Services

Web Service é um serviço prestado de um dispositivo eletrônico para outro dispositivo. Neste projeto, Web Service será referenciado aos serviços prestados por um servidor online às aplicações web e móveis, estabelecendo interligações entre elas e a base de dados, que está alojada no próprio servidor.

Um Web Service, utiliza tecnologia Web (neste projeto, foi utilizado o PHP) para estabelecer comunicação máquina-a-máquina, especificamente, para transferir ficheiros que sejam fáceis de ler ou decodificar por máquinas, utilizando linguagens padronizadas como o XML ou JSON.

A utilização de Web Services neste sistema, foi derivada de várias razões:

1. Interoperabilidade. Web Services são compatíveis com praticamente qualquer sistema ou plataforma, assim tendo uma longevidade e uma adaptação a mudanças do sistema muito maior que outras maneiras de comunicação de sistemas. Isto também permite a comunicação entre sistemas com linguagens diferentes. Neste projeto, temos a linguagem Java providente da aplicação móvel a comunicar com MySQL da base de dados, através dos Web Services.
2. Protocolos padronizados. Web Services usam linguagens padronizadas ao enviar informação para os dispositivos. Assim, reduzindo o tempo de desenvolvimento dos sistemas de leitura de dados e aumentando a qualidade do sistema em geral.
3. Comunicação low-cost. Ao utilizar tecnologia Web, os Web Services tornam-se num meio de comunicação muito barato de se implementar.

4.2. Desenvolvimento

Nesta secção, serão detalhados os passos necessários para a construção do sistema. A mesma inclui diagramas de hierarquia e apresenta os desafios de maior importância que tiveram de ser superados para a concretização do projeto, no seu todo. Esta secção encontra-se dividida em três partes, as quais compõem o sistema no seu todo: a aplicação web que serve para a gestão de dados para os administradores do sistema, a aplicação mobile que permite aos funcionários registar as suas presenças e uma subsecção, terceira parte, dedicada ao método de interligação entre as duas componentes de administração e registo de presenças.

4.2.1. Aplicação Web

4.2.1.1. Diagrama de hierarquia

O diagrama de hierarquia da aplicação web subdivide-se em duas componentes principais: A *Homepage* e o *Dashboard*. A *Homepage* destina-se aos visitantes e aos recursos que são comuns a todas as organizações que utilizam a aplicação, isto é, registo da organização, *download* da aplicação móvel, informações e contactos. O *Dashboard* é o conjunto de interfaces que permitem a gestão da aplicação para cada organização.

A *Homepage* é composta pelas seguintes interfaces:

1. Página inicial: É a interface inicial que contém uma introdução à aplicação WorkPresence e às suas funcionalidades principais.
2. *Downloads*: Contém o *link* de acesso que permite efetuar o download da aplicação móvel para os funcionários, e ainda os detalhes de como instalar a aplicação.
3. Registo para as organizações: Esta interface permite que cada organização efetue o seu registo inicial na aplicação.
4. Sobre: Contém informações, diversas, sobre a aplicação e os contactos de ajuda e suporte ao sistema.

Todas as interfaces apresentadas são composta por uma barra de navegação, que contém os links da *Homepage*, a interface de login que dá acesso às organizações ao *Dashboard* e um botão para a recuperação de *passwords*.

O *Dashboard* é constituído pelas seguintes:

1. Atividade dos funcionários: A página inicial do *Dashboard*, contém o registo das presenças mais recentes dos funcionários na organização. Nota: O administrador pode fazer check-in/out manual dos funcionários, sempre que se verificar alguma anomalia do sistema.
2. Departamentos: A interface Departamentos permite ao administrador gerir os departamentos da organização e alocar funcionários.
3. Funcionários: O administrador gere os dados relativos aos funcionários da organização (dados pessoais, departamento, horas).
4. Horários: O administrador gere os horários da organização e atribui os mesmos aos funcionários.
5. Geolocalização: Nesta interface, encontra-se um mapa com a localização mais recente de cada funcionário.
6. Estatísticas: Com recurso a esta interface é possível aceder a diversas estatísticas que possibilitam ilustrar a atividade dos funcionários. O administrador pode restringir a visualização das estatísticas de acordo com diferentes parâmetros (ex. datas específicas, departamentos).
7. Opções: Esta opção permite aos Administradores, configurar dados sobre a organização incluindo a gestão dos pontos de localização da organização.

Tal como na Homepage, o *Dashboard* contém uma barra de navegação, localizado no topo da página, com os vários links de acesso a todas as opções do sistema. O diagrama de hierarquia da aplicação web é apresentado na Figura 19.

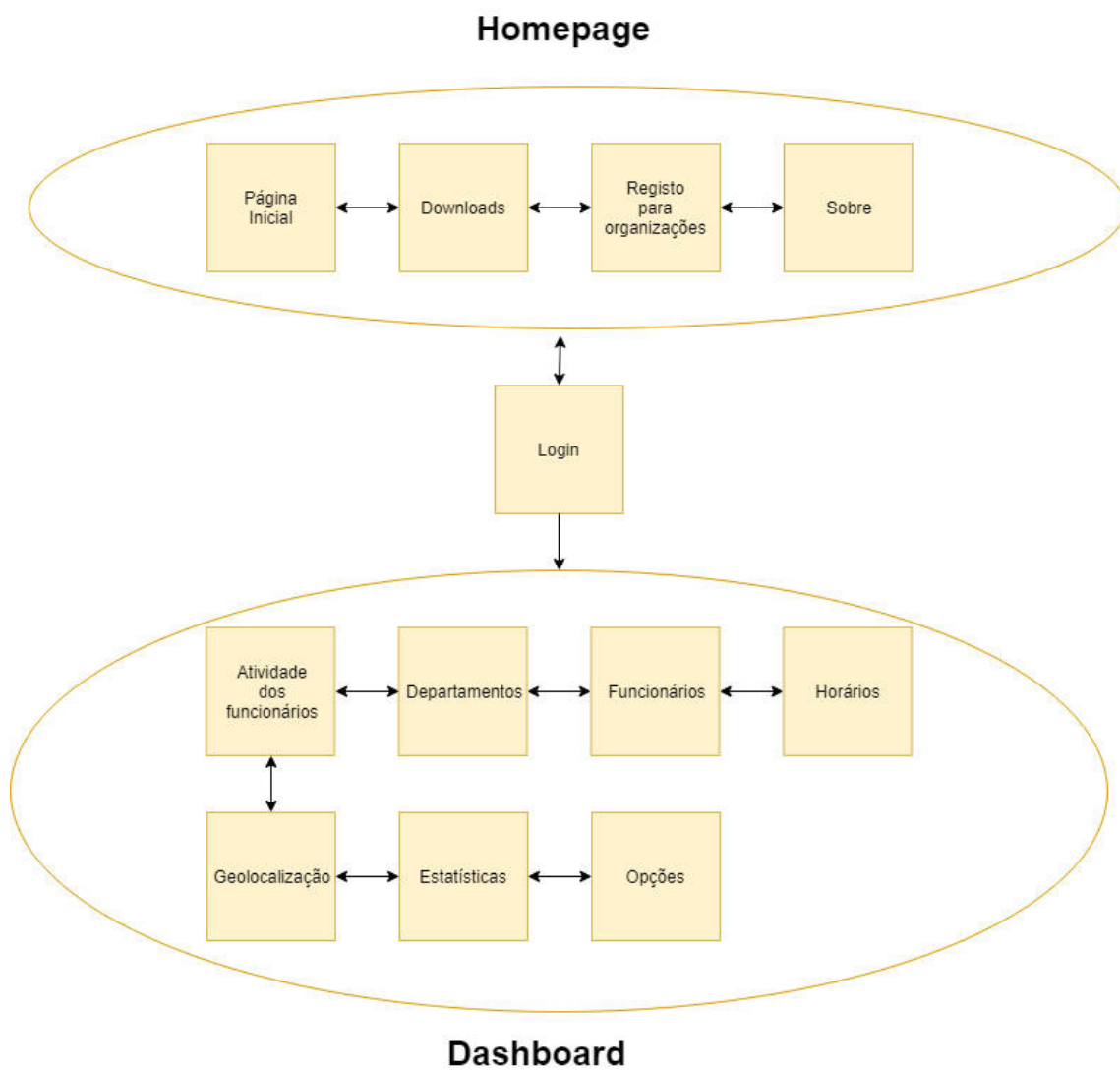


Figura 19 - Diagrama de hierarquia da aplicação web

O digrama da Figura 19 é o resultado de todas as interfaces mencionadas anteriormente.

4.2.1.2. Homepage

A *Homepage* é o conjunto de ecrãs disponíveis aos visitantes, ou seja, às organizações sem registo ou administradores registados que pretendem fazer login. É a parte do sistema mais informativa, em que as únicas funcionalidades que interagem com a base de dados são:

- Registo de organização
- Login de administrador
- Recuperação de *password*

A página principal do sistema é ilustrada na figura seguinte (Figura 20):

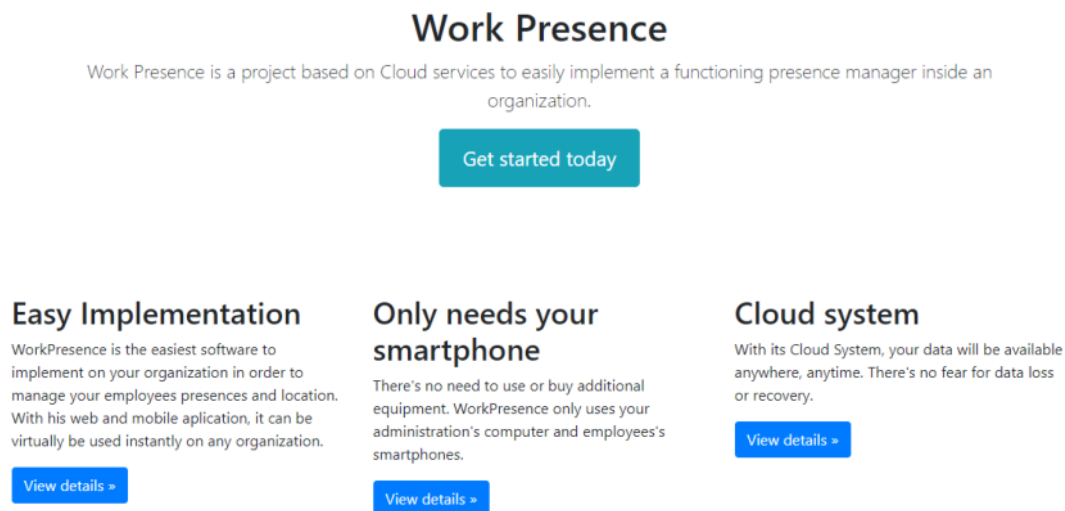


Figura 20 - Página inicial da aplicação web

Como está ilustrado na Figura 20, a página inicial descreve, sucintamente, as características principais do projeto. Ao clicar num dos botões de “Mostrar detalhes”, é mostrado um *pop-up*, com uma explicação mais detalhada de cada característica.

Também é de notar que existe uma barra de navegação no topo da página, que contém todos os *links* pertencentes à *Homepage* e campos para efetuar o *login* de administrador.

4.2.1.3. Dashboard

O *Dashboard*, componente que gere todas as funcionalidades administrativas de uma organização, é um dos componentes fundamentais do sistema, uma vez que este contém as funcionalidades principais que permitem o correto e essencial funcionamento de o sistema no seu todo.

Ainda que contendo diferentes interfaces, o *layout* foi desenhado e implementado com o intuito da sua utilização ser fácil e intuitiva. Esta contém uma barra de funcionalidades composta por botões que permitem aceder às diferentes ações relacionadas com o respetivo interface.

A interface adapta-se consoante o botão selecionado (Ex: ao clicar no botão “Editar Departamento”, aparecem os campos para selecionar e editar um departamento), sem que o utilizador necessite de mudar de página/janela. Assim, conseguimos ter várias funcionalidades numa única interface, sem sobrecarregar o espaço visual, criando deste modo uma utilização fácil e amigável.

4.2.1.3.1. Atividade dos funcionários

Quando o administrador acede a esta funcionalidade, atividade de funcionários, a aplicação apresenta uma tabela com a atividade mais recente dos funcionários daquela organização. Esta tabela é atualizada em tempo real, para que o administrador consiga ver, eficazmente, os seus funcionários a marcarem presenças. Também é possível pesquisar os funcionários pelo seu nome e por data da atividade. Todas estas funcionalidades podem ser vistas na Figura 21:

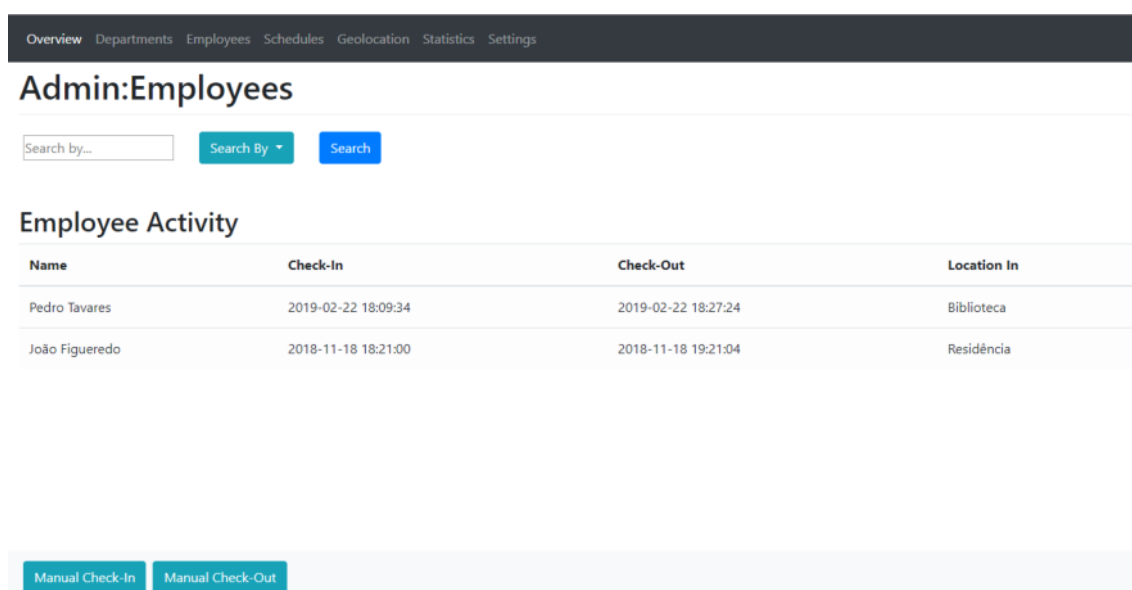


Figura 21 - Interface "Atividade Funcionários"

Na barra de funcionalidades desta interface, encontram-se dois botões:

1. Botão "Check-in Manual". Permite ao administrador fazer um check-in de um funcionário, caso ocorra alguma anomalia.
2. Botão "Check-out Manual". Permite ao administrador fazer um check-out de um funcionário.

Estes menus contêm restrições, de modo a que só se possa fazer check-in e check-out a funcionários válidos, isto é, para fazer um check-out a um funcionário, é necessário que a última ação do funcionário seja um check-in e vice-versa. Esta condição permite manter a integridade das sessões dos funcionários.

Adicionalmente, é também possível filtrar um funcionário, recorrendo à lista de funcionários de modo a ser mais fácil e intuitivo encontrar um determinado funcionário.

4.2.1.3.2. Departamentos

A interface dos departamentos permite gerir os departamentos de uma organização. Ao aceder a esta opção são apresentados todos os departamentos com os respetivos campos: nome de departamento, número de funcionários no departamento e data de criação do departamento. É possível mudar a ordem dos registos desta tabela alfabeticamente, por idade do departamento e pelo número de funcionários. Ao clicar num departamento da tabela, são mostrados todos os funcionários desse departamento de forma mais detalhada.

A barra de funcionalidades da interface “Departamentos”, contém:

1. “Criar Departamento”. Abre a interface, com os campos necessários, para criar um departamento.
2. “Editar Departamento”. Abre a interface que permite seleccionar e editar esse departamento.
3. “Eliminar departamento”. Abre a interface que permite seleccionar um departamento e posteriormente eliminá-lo, caso o mesmo não contenha funcionários alocados.

4.2.1.3.3. Funcionários

Ao abrir a interface dos funcionários, aparece uma tabela com todos os funcionários da organização e com os seguintes campos: nome, idade, email e “role”. Existem várias opções de filtração e ordenação tal como nas interfaces prévias. Ao clicar num funcionário da tabela, são exibidos os detalhes do perfil desse funcionário.

A barra de funcionalidades inclui:

1. “Criar funcionário”. Abre uma interface com os campos necessários para inserir um novo funcionário na organização.
2. “Editar funcionário”. Abre uma interface que permite seleccionar um funcionário e editar os seus campos e o seu departamento.
3. “Eliminar funcionário”. Abre uma interface que permite seleccionar um funcionário e eliminá-lo.

4.2.1.3.4. Horários

Esta interface permite gerir os horários de uma organização e atribuir funcionários aos mesmos. Ao abrir a interface “Horários” é mostrada uma tabela com todos os horários da organização, com o nome do horário e o número de funcionários que fazem esse horário. Ao clicar num horário da tabela, são mostrados os dias, horas e a lista de funcionários desse horário.

A barra de funcionalidades incluída nesta interface contém:

1. “Criar Horário”. Abre uma interface com os campos para criar um novo horário. Neste menu, o administrador atribui dias e horas ao horário.
2. “Editar Horário”. Permite editar o nome, ou os dias, ou as horas do horário selecionado.
3. “Apagar Horário”. Permite eliminar um horário definido.
4. “Atribuir Funcionários”. Abre uma interface que permite selecionar um horário e escolher os funcionários que pertencem/realizam esse horário.

4.2.1.3.5. Geolocalização

A interface “Geolocalização” não contém nenhuma barra de funcionalidades ao contrário das interfaces precedentes. A sua interface é apenas constituída por um único mapa onde aparecem as localizações da última atividade de cada funcionário.

Cada funcionário está assinalado no mapa com a sua foto de perfil para fácil identificação. Ao clicar numa foto de perfil no mapa são mostrados os detalhes do respetivo funcionário.

4.2.1.3.6. Estatísticas

A interface “Estatísticas” tem como principal função, calcular e mostrar ao administrador diferentes tipos de estatísticas de presenças, relativa aos funcionários da sua organização. No topo da interface, encontram-se apenas dois campos que permitem seleccionar datas, apresentando as estatísticas de acordo com as duas datas especificadas. No caso de o utilizador não colocar informação, datas, nestes campos a aplicação remete as estatísticas gerais de presenças.

A barra de funcionalidades é composta por dois botões:

1. “Por Funcionários”. Mostra as estatísticas relevantes a cada funcionário, que inclui as horas totais e a média de horas realizada por dia. Ao clicar num funcionário, da lista, é apresentado um gráfico com as horas que ele fez por dia.
2. “Por Departamentos”. Apresenta as estatísticas, consideradas relevantes, em cada departamento, incluindo as horas totais de todos os funcionários de cada departamento e a média de horas feitas por cada funcionário do departamento. Ao clicar num departamento da lista, aparece um gráfico de linhas com as horas de cada funcionário do departamento, em que cada linha corresponde a um funcionário. As estatísticas de departamento também incluem um botão que permite criar um gráfico circular, com as horas totais de todos os departamentos. Esta opção permite, criar uma visualização mais sucinta e rápida do número de horas registado em cada departamento.

4.2.1.3.7. Opções

A interface “Opções” contém as funcionalidades relacionadas com a manipulação de dados da própria organização. Estas funcionalidades, que se encontram na barra de funcionalidades, são:

1. “Editar organização”. Permite editar os dados da organização, tais como o nome da organização, o supervisor, o contacto e o email da organização.
2. “Adicionar ponto de localização”. Abre um mapa que permite ao administrador seleccionar um ponto geográfico, ou vários pontos

geográficos, e registá-lo como ponto de entrada para os funcionários. Estes pontos de entrada restringem onde os funcionários podem fazer check-in, para que não haja presenças registradas fora do local de trabalho. O administrador da organização também pode dar nomes a estes pontos de entrada. Assim, em vez de aparecerem as coordenadas dos pontos geográficos, aparecem os nomes, facilitando a visualização da atividade dos funcionários.

3. “Remover ponto de localização”. Permite remover um ponto de localização definido previamente na organização.
4. “Ver pontos de localização”. Abre uma lista com todos os pontos de localização da organização.

4.2.1.4. Funções em destaque

Esta secção é dedicada às pequenas bibliotecas criadas somente pelo autor. Apesar de a maior parte da aplicação Web ser construída com recurso a funções de outras bibliotecas (jQuery e Bootstrap), foi também necessário desenvolver novas funções, de modo a complementar as bibliotecas em uso, que permitissem realizar algumas ações muito específicas do projeto. Apresentamos algumas dessas funções:

4.2.1.4.1. *FillTable*

Esta função escrita em Javascript é utilizada em quase todas as interfaces do *Dashboard*. Esta função recebe as seguintes variáveis: *headings*, *content* e *id*. O objetivo desta função é criar uma tabela HTML com as variáveis recebidas. O código de *FillTable* encontra-se no ANEXO A1.

A variável *headings* contém os títulos correspondentes a cada coluna da tabela organizados dentro de um *array*. Estes títulos formam a primeira linha da tabela e destacam-se a negrito. O tamanho desta variável determina o número de colunas que a tabela irá formar.

A variável *content* determina o conteúdo da tabela. *Content* é um *array* de *arrays*. Cada elemento em *content* é um *array* que corresponde a cada título da variável *headings* na mesma posição. Ou seja, se o primeiro elemento de *headings* for “Nome de funcionário”, então o primeiro elemento de *content* será um *array* com todos os nomes dos funcionários. Assim, para o correto funcionamento da função, o tamanho de *headings* tem de ser igual ao tamanho de *content* e todos os elementos dentro de *content* têm de ter igual tamanho, de modo a determinar o número de linhas da tabela.

A variável *id* indica o local na página HTML a colocar a tabela resultante da função.

Como exemplo, se tivermos as seguintes variáveis:

- *Headings*: [Name,Check-In]
- *Content*: [
[Pedro Tavares,João Figueredo],
[2019-02-22 18:09:34, 2018-11-18 18:21:00]
]

Obtemos o seguinte resultado, visualizado na Figura 22:

Name	Check-In
Pedro Tavares	2019-02-22 18:09:34
João Figueredo	2018-11-18 18:21:00

Figura 22 - Exemplo da função "FillTable"

Esta função foi feita tendo em conta o formato de dados recebidos da base de dados. Como os dados são recebidos através de *arrays* (formato JSON) e ordenados pelos atributos da base de dados (nome, idade, etc...), facilmente se cria um *array* com estes dados, atribui-se títulos a cada atributo e constrói-se uma tabela para ser visualizada facilmente e eficazmente.

Outra versão desta função é *FillTableExtra*, que contém uma variável adicional chamada *extra*. Esta variável é um *array* com informação a colocar nos elementos “<tr>” de cada linha da tabela HTML. Isto é particularmente útil quando se quer modificar cada linha da tabela, especificamente, para mostrar informação ao utilizador quando clica numa linha da tabela.

4.3. Aplicação móvel

4.3.1. Diagrama de Hierarquia

O diagrama de hierarquia da aplicação móvel foca-se numa única *Dashboard* para os funcionários, onde se encontram todas as funcionalidades a que os funcionários podem aceder. O diagrama de hierarquia é apresentado na figura seguinte (Figura 23):

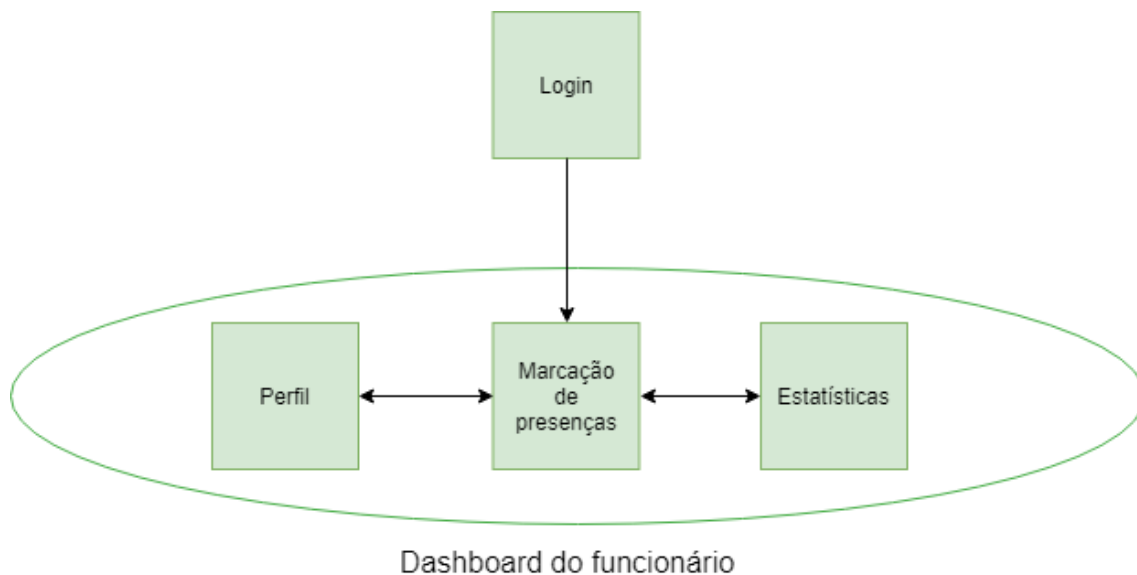


Figura 23 - Diagrama de hierarquia da aplicação móvel

Ao abrir a aplicação, é apresentado a interface de login, onde o funcionário coloca as suas credenciais para aceder ao seu *Dashboard* de funcionário. O *Dashboard* de funcionário é composto por:

1. Marcação de presenças. Esta é a interface principal, que aparece imediatamente ao login bem-sucedido do funcionário. Ao abrir a interface, é mostrado um mapa com a localização atual do utilizador, juntamente com marcadores a indicar os pontos de entrada onde o funcionário pode fazer check-in/out. No fundo da interface, encontra-se um botão que permite ao funcionário fazer check-in ou check-out dependendo da sua última presença. Este interface é ilustrada na Figura 24:

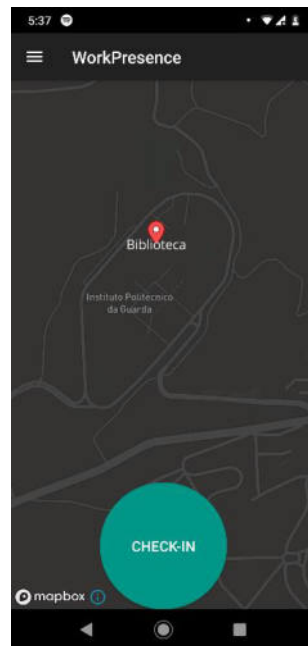


Figura 24 - Interface "Marcação de presenças"

2. Estatísticas. Esta interface apresenta o histórico de presenças do funcionário, com a data e horas feitas em cada sessão. O funcionário pode escolher duas datas e filtrar as suas presenças entre as duas datas escolhidas.
3. Perfil. Mostra os dados pessoais do utilizador incluindo o departamento e o horário atual do funcionário.

4.3.2. Navegação entre interfaces

De modo a obter uma navegação eficiente e intuitiva entre as interfaces do *Dashboard* do funcionário, foi implementado um *NavigationDrawer* na aplicação móvel.

NavigationDrawer é um painel que está escondido por omissão, mas ao deslizar para a direita, no ecrã do *smartphone*, o painel aparece. O painel é constituído por vários botões, quando clicados, que mudam a interface principal. Este painel pode ser visualizado no ANEXO B1.

Adicionalmente, foram implementadas classes *Fragment*, em vez das habituais *Activities* (*Activities* são as classes normalmente utilizadas para cada interface individual). As classes *Fragment* são interfaces que conseguem ser substituídas, sem a necessidade de recarregar o ecrã. Isto permite uma maior fluidez e rapidez ao utilizar a aplicação móvel.

Ao colocar um elemento `<fragment>` no *layout* da aplicação, podemos alternar entre várias classes *Fragment* e atualizar o elemento entre elas, sem a necessidade de sair do *layout*. A Figura 25, detalha este processo:

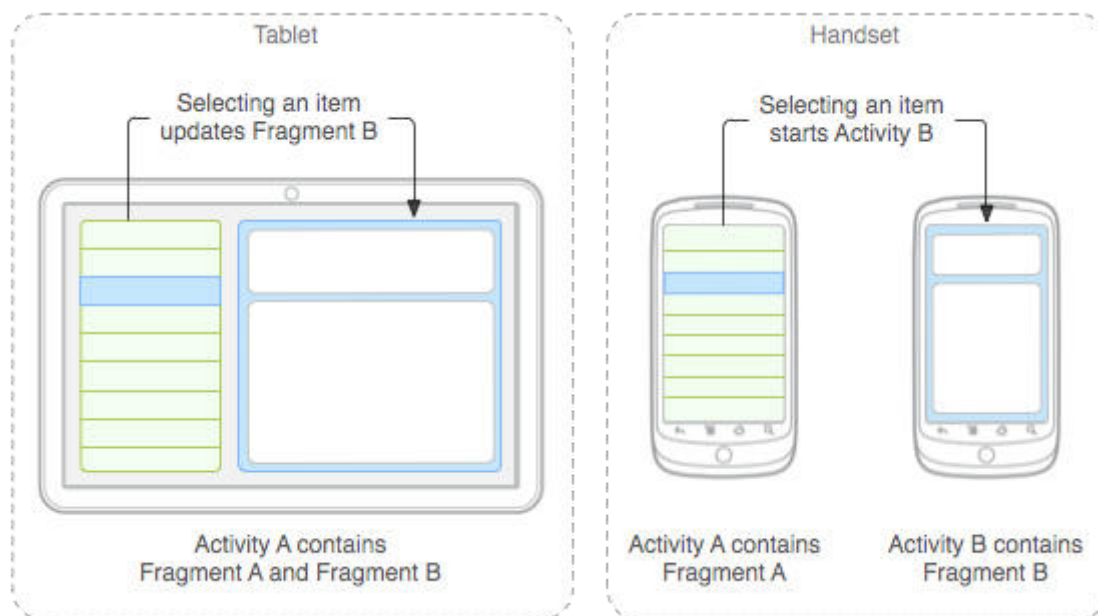


Figura 25 - Diferenças entre Activities e Fragments [29]

Como a Figura 25 detalha no primeiro exemplo, o ecrã do *tablet* é atualizado com o *Fragment B* sem mudar de ecrã ou sem destruir a *Activity* em que se encontra. No segundo exemplo, quando o utilizador seleciona a *Activity B*, a aplicação é forçada a começar uma nova *Activity*, gastando mais recursos de memória e tornando a aplicação menos fluida para o utilizador.

4.3.3. Interligações do sistema

Nesta subsecção, serão discutidas as interligações entre os vários componentes do sistema: aplicação web, aplicação móvel e a base de dados. Estas interligações são feitas através de Web Services alojados num servidor online. As ligações do sistema encontram-se na seguinte figura (Figura 26):

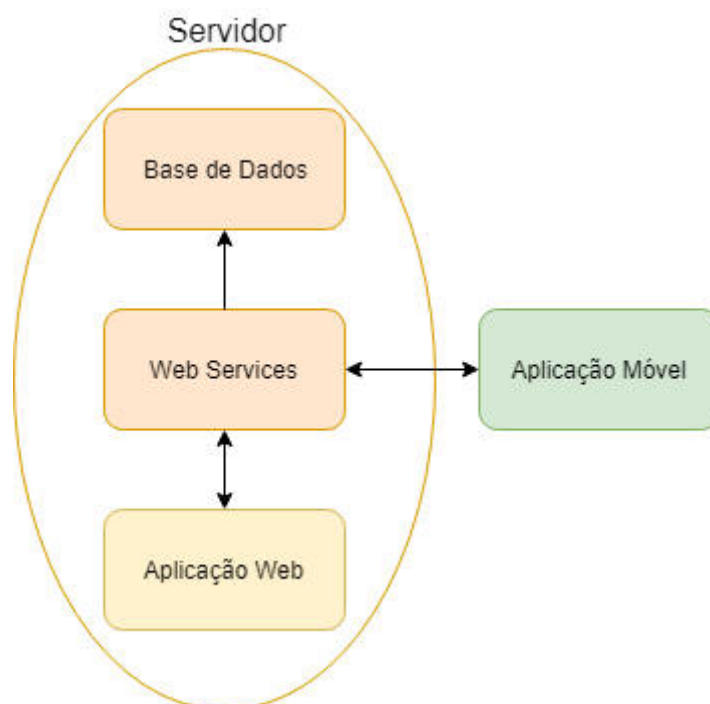


Figura 26 - Ligações entre os vários componentes do sistema

Como podemos concluir da figura acima (Figura 26), o sistema é composto por um servidor online que contém a base de dados, Web Services e a aplicação Web. Quer a aplicação móvel, quer a aplicação Web, conseguem receber e enviar dados para a base de dados através dos WebServices.

Neste projeto, cada *WebService* individual divide-se em duas partes que permitem:

1. Estabelecer uma conexão à base de dados, chamar um procedimento da base de dados e obter os dados desse procedimento.
2. Processar os dados obtidos e transformá-los em formato JSON, para depois serem lidos por outra aplicação.

Assim, para cada procedimento da base de dados, corresponde um *WebService*, providenciando uma adaptação muito maior a mudanças do sistema. Pelo que sempre que é preciso alterar o código de um procedimento, conseguimos restringir essa mudança à base de dados e ao *WebService* que contém esse procedimento.

No entanto, este método de programar *WebServices* provoca uma exponenciação do número de ficheiros do servidor (atualmente são 51 ficheiros *Webservice*) e, conseqüentemente, uma maior carga no servidor devido ao maior número de pedidos *WebService*.

Chegou-se à conclusão que seria mais benéfico ter uma melhor adaptação a mudanças do sistema, para diminuir o tempo de desenvolvimento do *software* futuro. A carga de processamento que o servidor terá, não deve ser significativa o suficiente para desenvolver em torno dela. Adicionalmente, se chegar a um ponto em que o servidor se torne muito lento, é muito simples aumentar o seu processamento no corrente estado tecnológico dos servidores *Web*.

4.4. Testes

Esta secção é dedicada a testes realizados, de modo a assegurar a qualidade de *software*, e considerados significativos para o bom funcionamento do sistema. Não obstante, refere-se o facto de que ao longo de todo o processo de desenvolvimento se terem realizado diferentes tipos de testes funcionais como são exemplo a validação de campos, validação de ações de botões entre outros.

4.4.1. Cálculo da distância entre pontos geográficos

Para registar um *check-in*, é necessário o funcionário estar dentro do alcance do ponto de entrada. Assim, é preciso calcular a distância entre dois pontos geográficos (o ponto em que o funcionário se encontra e entre todos os pontos de entrada). Nesta subsecção, é explicado como o algoritmo deve funcionar e são apresentados alguns resultados obtidos após realização dos testes.

Esta verificação da localização do funcionário é feita através de uma função que se encontra na base de dados, chamada *VerDistânciaPontos*. O código desta função pode ser visualizada no ANEXO C1. Esta função recebe três variáveis: *lat* (latitude), *log* (longitude) e *ido* (id da organização).

O objetivo desta função é verificar se o ponto geográfico recebido se encontra dentro de um dos vários pontos de entrada da organização, cujo *id* é o mesmo que a variável *ido*. Para isso, foi criado um algoritmo com os vários passos a serem efetuados:

1. Determinar a distância entre o ponto geográfico recebido e a localização geográfica dos pontos de entrada da organização.
2. Comparar essa distância com o alcance dos pontos geográficos. Se a distância calculada for menor do que o alcance, então o utilizador encontra-se dentro do ponto de entrada.
3. Devolver o id do ponto de entrada em que se encontra.
4. Se não encontrar ponto de entrada válido, devolve *null*.

Para determinar a distância entre dois pontos geográficos, é utilizada a seguinte função:

```
111.111 *
  DEGREES(ACOS(COS(RADIANS(latitude1))
    * COS(RADIANS(latitude2))
    * COS(RADIANS(longitude1 - longitude2))
  + SIN(RADIANS(latitude1))
    * SIN(RADIANS(latitude2))))
```

A função anterior calcula a distância entre o primeiro ponto geográfico (latitude1, longitude1), que representa a localização atual do funcionário, e o segundo ponto geográfico (latitude2, longitude2), que apresenta um ponto de entrada. Comparando o valor resultante desta função com o alcance do ponto de entrada, conseguimos determinar se o funcionário se encontra dentro do ponto de entrada.

Para testar esta função, foram criados dois pontos de entrada:

1. Ponto de entrada “Ação Social”, com as coordenadas “-7.270594741136932,40.53554774828032”, com um alcance de 50 metros e com o *id* 1.
2. Ponto de entrada “Biblioteca”, com as coordenadas “-7.270594741136932,40.53554774828032”, com um alcance de 100 metros e com o *id* 2.

Se definirmos o ponto geográfico do funcionário dentro do ponto de entrada “Ação Social”, o resultado obtido será “1”. As coordenadas escolhidas foram: “40.535684, -7.270628”. Ao executar a seguinte função:

```
Select VerDistanciaPontos( -7.270628,40.535684,1);
```

Obtemos o resultado na Figura 27:

```
VerDistanciaPontos( -7.270628,40.535684,1) ÷
1
```

Figura 27 - Resultado do primeiro teste da função "VerDistanciaPontos"

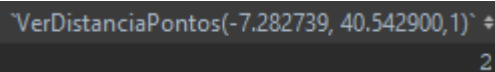
Como podemos ver, o resultado é igual ao esperado e indica que o ponto geográfico indicado se encontra dentro do ponto de entrada com o *id* 1, chamado “Ação Social”.

No segundo teste, vamos definir o ponto de geográfico do funcionário dentro do ponto de entrada “Biblioteca”, utilizando as coordenadas: “-7.282739, 40.542900”. Assim, o resultado deverá ser o id do ponto de entrada “Biblioteca”: 2.

Ao executar a função:

```
Select VerDistanciaPontos(-7.282739, 40.542900,1);
```

Obtemos o resultado na Figura 28:



The screenshot shows a dark-themed interface with a query result. The text displayed is: `VerDistanciaPontos(-7.282739, 40.542900,1) =` followed by the value `2` on a new line.

Figura 28 - Resultado do segundo teste da função "VerDistanciaPontos"

O resultado é igual ao esperado, comprovando que a função está a funcionar corretamente e que consegue identificar se o funcionário está dentro de um ponto de entrada da organização.

Estes testes permitiram verificar a eficácia das funções testadas. Assim, conseguimos detetar erros (se existirem) e garantir a integridade do sistema. Se as funções não são testadas devidamente, então não existe maneira de comprovar se o sistema funciona como suposto.

5. CONCLUSÕES

Este projeto foi uma oportunidade para alargar a minha experiência e o meu conhecimento de análise de requisitos, desenvolvimento de *software* e programação de sistemas móveis, através da aplicação dos conhecimentos adquiridos na licenciatura e na parte académica do mestrado. O desenvolvimento do projeto, WorkPresence, envolveu a aprendizagem e o uso de distintas tecnologias: Web; mobile; base de dados e Webservice, que me permitiram aprofundar os meus conhecimentos. O produto final deste projeto encontra-se no *website* “workpresence.online” até à sua apresentação.

Aquando da proposta inicial foi nos proposto o desenvolvimento de um sistema, utilizando tecnologias mobile, para registar e gerir as presenças dos funcionários de uma organização. Foi estabelecido, como objetivo, que o sistema deveria constar de uma aplicação Web para os administradores gerirem a sua organização no seu todo (funcionários, departamentos, horários) e uma aplicação móvel que permitisse o registo de entrada e saída dos funcionários (registo de presenças). Podemos assim afirmar que todos os objetivos propostos, inicialmente, foram concluídos com sucesso. De modo sucinto apresentam-se as funcionalidades desenvolvidas ao longo do projeto:

A Aplicação móvel permite:

- Registo de login dos funcionários.
- Registo de check-in/check-out do funcionário.
- Visualizar horário de trabalho do funcionário.
- Analisar estatísticas referentes às horas de trabalho do funcionário.

A Aplicação Web permite:

- Gerir funcionários e respetivos departamentos afetos aos funcionários.
- Visualizar presenças dos funcionários.
- Gerir horários de trabalho para os funcionários.
- Analisar estatísticas de trabalho de cada funcionário e departamento.

No entanto, apesar de o projeto estar finalizado num protótipo que consideramos fiável, existem ainda alguns aspetos possíveis de melhorar, que passamos a descrever:

- Um *design* mais limpo, que comunica mais eficazmente com o utilizador, onde o espaço visual utilizado é mínimo para mostrar os dados necessários ao utilizador.
- Criar uma aplicação Web, para os funcionários, de modo a que possam marcar presenças através do computador.
- Implementação de utilidades pequenas, por exemplo, guardar as credenciais de login no dispositivo.
- Personalização das aplicações para cada organização, como a possibilidade de colocar as cores da organização e o logótipo dentro das aplicações.
- Implementação de estatísticas e notificações de atrasos dos funcionários.
- Check-in automático de um funcionário, quando estiver no alcance de um ponto de entrada, durante a sua hora de entrada.
- Obter certificados de segurança para a proteção de dados.
- Expansão da aplicação para um mercado global, ou seja, tornar a aplicação disponível e operacional para qualquer utilizador e organização.

A razão pela qual não foi possível implementar estas funcionalidades, prende-se com o facto de que foi necessário dedicar muito tempo para implementar o sistema em si, nomeadamente no que diz respeito à comunicação entre a base de dados e as aplicações Web e mobile, bem como a programação e validação de todos os controlos necessários que permitiram realizar um sistema, Workpresence, de qualidade.

Apesar de estas funcionalidades não estarem contempladas nesta versão do sistema Workpresence, podemos afirmar que a aplicação se encontra funcional e permite que os seus utilizadores possam melhorar a gestão de presenças nas organizações. Com o seu eventual melhoramento, especificamente, certificados de segurança de dados e implementação de novas funcionalidades, é possível tornar esta aplicação com uma maior qualidade, permitindo uma distribuição do sistema para o mercado global.

Este projeto consegue tornar a gestão de presenças dos funcionários e a sua análise, num processo praticamente automatizado, no qual, os administradores só necessitam de introduzir os dados da sua organização.

6. REFERÊNCIAS BIBLIOGRÁFICAS

- [1] “The Importance of Employee Attendance Reporting,” Mitrefinch, [Online]. Available: <https://mitrefinch.com/blog/importance-employee-attendance-reporting/>. [Acedido em 11 11 2017].
- [2] S. Fletcher, “TEMPO,” [Online]. Available: <https://www.tempo.io/blog/the-importance-of-employee-time-tracking>. [Acedido em 24 04 2019].
- [3] E. Araújo, “Run - Disertação de mestrado,” Lisboa, 2011.
- [4] E. Mordini e C. Petrini, “Ethical and social implications of biometric identification technology,” Istituto Superiore di Sanità, Roma, 2007.
- [5] “Proteção de dados,” PL & DUTEC, [Online]. Available: <https://protecao-dados.pt/o-regulamento/>. [Acedido em 01 06 2018].
- [6] M. Saias, “O Novo Regulamento Geral de Proteção de Dados Pessoais,” Raposo, Sá Miranda e Associados, [Online]. Available: <https://www.pra.pt/pt/communication/news/o-novo-regulamento-geral-de-protecao-de-dados-pessoais-em-3-minutos/>. [Acedido em 17 11 2017].
- [7] “CucoCloud,” NameIT, [Online]. Available: <https://cucocloud.pt/>. [Acedido em 11 11 2017].
- [8] “Attendance Manager,” [Online]. Available: <https://play.google.com/store/apps/details?id=com.yashketkar.attendancemanager&hl=en>. [Acedido em 10 10 2017].
- [9] “SynchroTeam,” SynchroTeam, [Online]. Available: <https://www.synchroteam.com/es/mobile.php>. [Acedido em 10 10 2017].
- [10] K. W. Collier, “What is a self-organizing team?,” em *Agile Analytics: A Value-Driven Approach to Business Intelligence and Data Warehousing*, Pearson Education, 2011.
- [11] A. Moran, *Agile Risk Management*, Springer Verlag, 2014.
- [12] P. Abrahamson, O. Salo, J. Ronkainen e J. Warsta, *Agile software development methods: Review and analysis*, VTT, 2002.
- [13] E. Gamma e K. Beck, “Extreme Programming,” [Online]. Available: <https://www.cs.usfca.edu/~parrr/course/601/lectures/xp.html>. [Acedido em 08 12 2017].
- [14] K. Beck, *Extreme Programming Explained: Embrace Change*, Addison-Wesley, 1999.
- [15] C. Tristram, “Everyone's a Programmer,” *Technology Review*, p. 39, Novembro 2003.

-
- [16] “Extreme Programming,” Dezembro 2011. [Online]. Available:
] <https://www.computerworld.com/article/2585634/app-development/extreme-programming.html>. [Acedido em 10 12 2018].
- [17] L. Araújo, “A Importância do teste de software para a qualidade do software,”
] Scribd.
- [18] G. Booch, J. Rumbaugh e I. Jacobson, Unified Modeling Language User Guide,
] The, 2nd Edition, Addison-Wesley Professional, 2005.
- [19] “The Importance of Using UML for Modeling,” W3Computing, [Online].
] Available: <http://www.w3computing.com/systemsanalysis/importance-using-uml-modeling/>. [Acedido em 5 12 2017].
- [20] M. K. Choubey, IT Infrastructure and Management, Pearson India: Pearson, 2011.
]
- [21] “UML diagrams,” uml-diagrams.org, 2009. [Online]. Available: <https://www.uml-diagrams.org/use-case-diagrams.html>. [Acedido em 07 01 2019].
- [22] “Use Cases,” usability.gov, [Online]. Available: <https://www.usability.gov/how-to-and-tools/methods/use-cases.html>. [Acedido em 8 1 2019].
- [23] S. W. Ambler, “UML 2 CLASS DIAGRAMS,” Ambyssoft Inc., [Online].
] Available: <http://www.agilemodeling.com/artifacts/classDiagram.htm>. [Acedido em 19 01 2019].
- [24] MDN, “HTML,” Mozilla, [Online]. Available: <https://developer.mozilla.org/en-US/docs/Web/HTML..> [Acedido em 22 09 2018].
- [25] “About the CSS 2.1 Specification,” W3, [Online]. Available:
] <https://www.w3.org/TR/2011/REC-CSS2-20110607/about.html>. [Acedido em 22 09 2018].
- [26] “jQuery,” The jQuery Foundation, [Online]. Available: <https://jquery.com/>.
] [Acedido em 2018 02 03].
- [27] “Why MySQL,” Oracle, [Online]. Available: <https://www.mysql.com/why-mysql/>.
] [Acedido em 2019 02 03].
- [28] “Statista,” Statista, [Online]. Available:
] <https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/>. [Acedido em 2019 02 07].
- [29] “Fragments,” Google, [Online]. Available:
] <https://developer.android.com/guide/components/fragments>. [Acedido em 2019 03 27].
-

ANEXOS

ANEXO A1

```
function FillTable(headings,content,id){
    try {
        nColumns = headings.length;
        nLines = content[0].length;
        result = "<thead><tr>";
        for (i = 0; i < nColumns; i++) {
            result += "<th>" + headings[i] + "</th>";
        }
        result += "</tr></thead><tbody>";
        for (i = 0; i < nLines; i++) {
            result += "<tr>";
            for (j = 0; j < nColumns; j++) {
                result += "<td>" + content[j][i] + "</td>";
            }
            result += "</tr>";
        }
        result += "</tbody>";
        document.getElementById(id).innerHTML = result;
    } catch (err) {
        document.getElementById(id).innerHTML = err;
    }
}
```

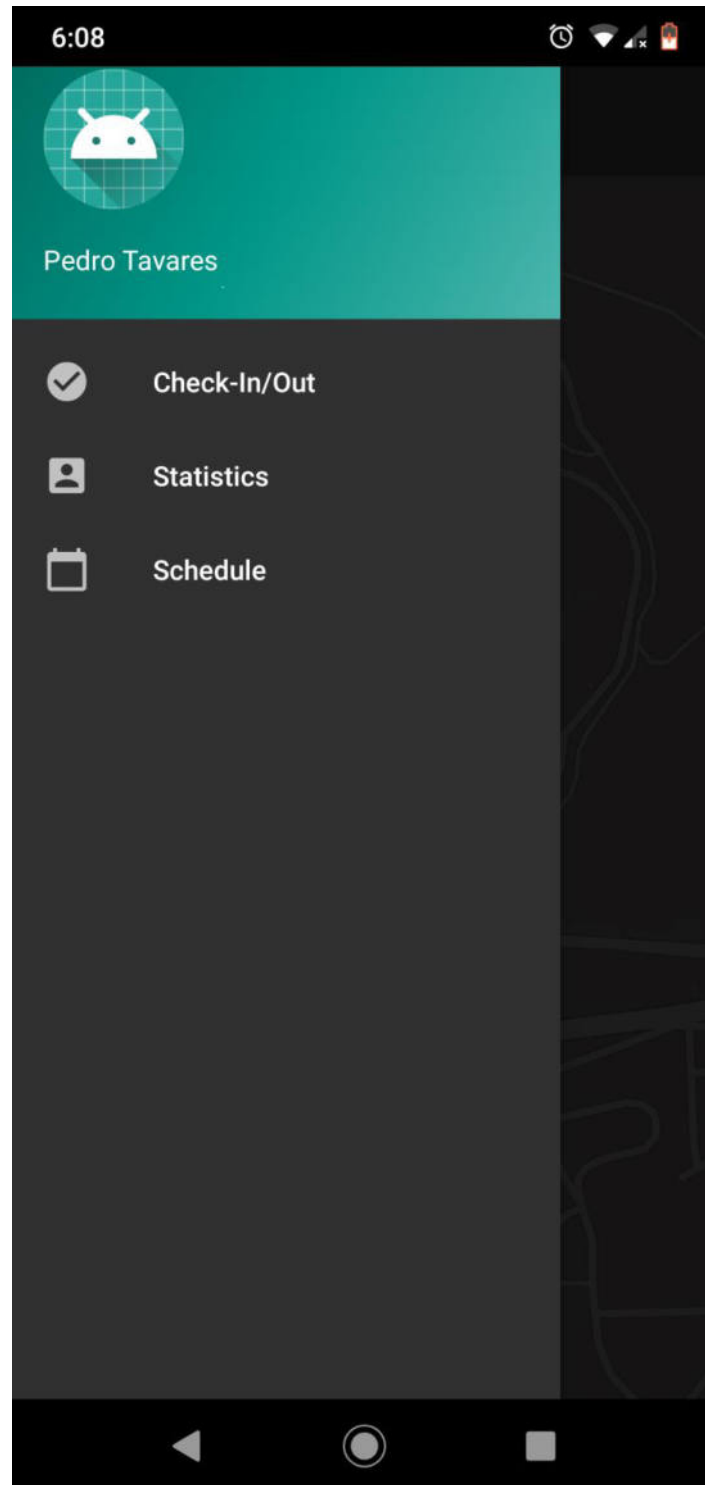
ANEXO A2

```
function GraphBarras(labelsAux, title, graphData, idCanvas, idContainer) {
    $("#"+idCanvas).remove();
    $("#"+idContainer).append("<canvas id='"+idCanvas+"' style='max-width: 500px;'></canvas>");
    var ctx = document.getElementById(idCanvas).getContext('2d');
    bC = ['rgba(255, 99, 132, 0.2)', 'rgba(54, 162, 235, 0.2)', 'rgba(255, 206, 86, 0.2)',
        'rgba(75, 192, 192, 0.2)', 'rgba(153, 102, 255, 0.2)', 'rgba(255, 159, 64, 0.2)'];
    borC = ['rgba(255, 99, 132, 1)', 'rgba(54, 162, 235, 1)', 'rgba(255, 206, 86, 1)',
        'rgba(75, 192, 192, 1)', 'rgba(153, 102, 255, 1)', 'rgba(255, 159, 64, 1)'];
    auxBC = []; auxBorC = []; j = 0;
    for (i = 0; i < graphData.length; i++) {
        auxBC[i] = bC[j];
        auxBorC[i] = borC[j];
        j++;
        if (j === 6) {
            j = 0;
        }
    }
    var myChart = new Chart(ctx, {
        type: 'bar',
        data: {
            labels: labelsAux,
            datasets: [{
                label: title,
                data: graphData,
                backgroundColor: auxBC,
                borderColor: auxBorC,
                borderWidth: 1
            }]
        },
        options: {
            responsive: true,
            scales: {
                yAxes: [{
                    ticks: {
                        beginAtZero: true
                    }
                }]
            }
        }
    });
}
```


ANEXO A3

```
function GraphPie(labels,graphData,idCanvas,idContainer){
    $("#"+idCanvas).remove();
    $("#"+idContainer).append("<canvas id='"+idCanvas+"' style='max-width: 500px;'></canvas>");
    var ctxP = document.getElementById("pieChart").getContext('2d');
    bC = [
        "#F7464A", "#46B8BD", "#FDB462", "#949FB1", "#4D5360"
    ];
    borC = [
        "#FF5A5E", "#5AD3D1", "#FFC870", "#A8B3C5", "#616774"
    ];
    auxBC = [];
    auxBorC = [];
    j = 0;
    for (i = 0; i < graphData.length; i++) {
        auxBC[i] = bC[j];
        auxBorC[i] = borC[j];
        j++;
        if (j === 5) {
            j = 0;
        }
    }
    var myPieChart = new Chart(ctxP, {
        type: 'pie',
        data: {
            labels: labels,
            datasets: [
                {
                    data: graphData,
                    backgroundColor: auxBC,
                    hoverBackgroundColor: auxBorC
                }
            ]
        },
        options: {
            responsive: true
        }
    });
}
```

ANEXO B1



ANEXO C1

```
create function VerDistanciaPontos(lat varchar(255), log varchar(255), ido int)
returns double
BEGIN
  DECLARE res int(11);
  SELECT
    p.idpontosorganizacoes
  into res
  FROM pontosorganizacoes AS p
  where p.Alcance > (111.111 *
    DEGREES(ACOS(COS(RADIANS(lat))
      * COS(RADIANS(p.latitude))
      * COS(RADIANS(log - p.longitude))
      + SIN(RADIANS(lat))
      * SIN(RADIANS(p.latitude)))) * 1000)
    and organizacoes_idOrganizacoes=ido
  LIMIT 1;
  RETURN res;
end;
```

ANEXO D1

Diagramas de casos de uso e sequência

6.1.1.1.1. Checkout

Nome	Checkout
Objetivo	Marcar a presença do utilizador
Atores	Funcionário
Pré-condição	Login efetuado e Check-in efetuado
Prioridade	Alta
Fluxo Principal	<ol style="list-style-type: none"> 1. O funcionário clica no botão “Checkout” na interface principal. 2. O sistema verifica a localização do dispositivo. 3. O sistema fecha a sessão e regista a hora, data, utilizador e localização atual. 4. O sistema apresenta mensagem de confirmação.
Fluxo Secundários	<ol style="list-style-type: none"> 2. A) O sistema verifica que o dispositivo não tem geolocalização ativa e termina a operação com uma mensagem de erro.
Exceções	<ol style="list-style-type: none"> 3. A) Não existe ligação à base de dados e o sistema cancela a operação.
Pós-condição	O botão “Checkout” altera-se para o botão “Check-In”.
Casos de teste	<ol style="list-style-type: none"> 2. Verificar se o sistema deteta a geolocalização efetivamente.

Tabela 16 - Descrição de caso de uso "Checkout"

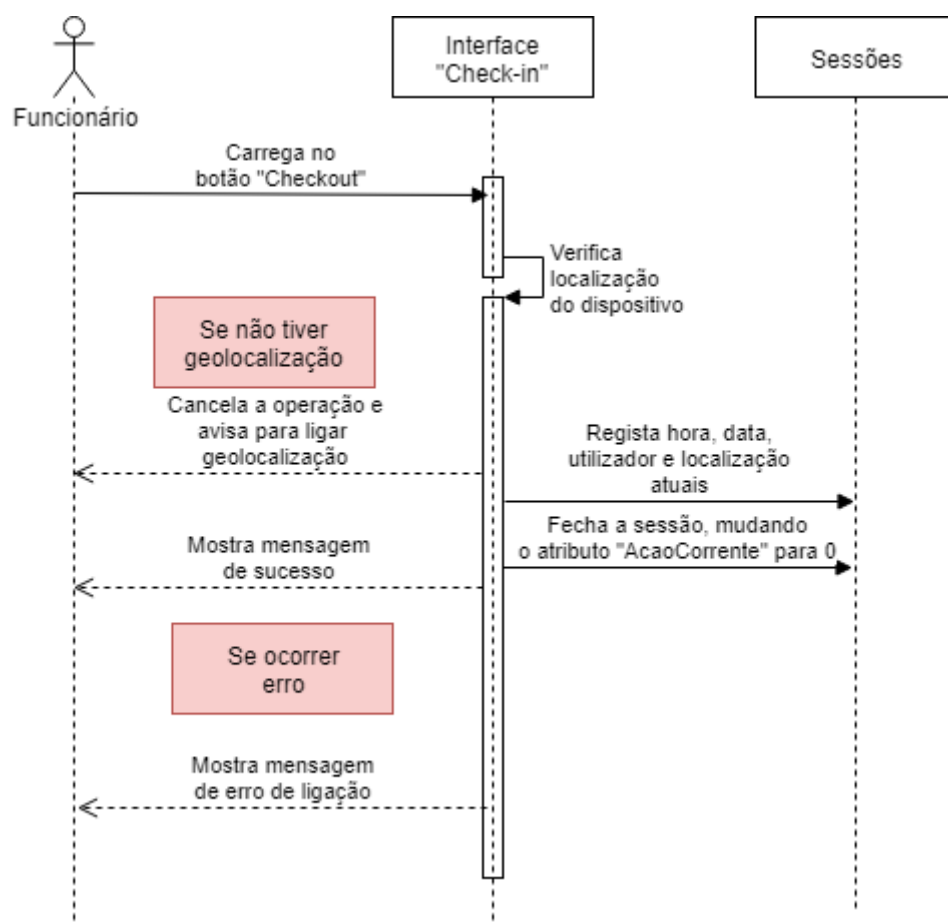
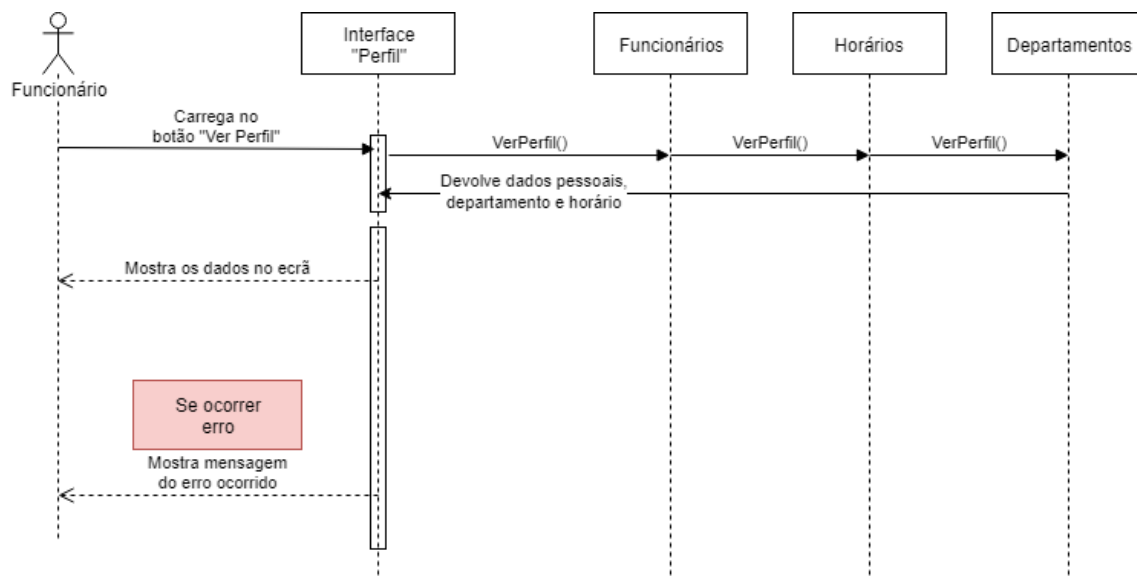


Figura 29 - Diagrama de sequência do "Checkout"

Ver perfil

Nome	Ver perfil
Objetivo	Mostrar os dados pessoais associados ao funcionário
Atores	Funcionário
Pré-condição	Login efetuado
Prioridade	Baixa
Fluxo Principal	<ol style="list-style-type: none"> 1. O funcionário acede à interface “Perfil”. 2. O sistema mostra os dados pessoais do funcionário, departamento e horário.
Fluxo Secundários	<ol style="list-style-type: none"> 2. A) Se ocorrer um erro, o sistema mostra uma mensagem do erro que ocorreu.
Exceções	
Pós-condição	
Casos de teste	

Tabela 17 - Descrição de caso de uso "Ver perfil"**Figura 30 - Diagrama de sequência "Ver Perfil"**

Gerir Horários

Nome	Gerir funcionários
Objetivo	Visualizar, inserir, editar e apagar horários
Atores	Administrador
Pré-condição	Login de administrador efetuado
Prioridade	Baixa
Fluxo Principal	<ol style="list-style-type: none"> 1. O administrador acede à interface “Horários”. 2. O sistema mostra os dados dos funcionários da organização do administrador e os botões que abrem os menus das outras funcionalidades (inserir, editar e apagar.). 3. O administrador clica no botão cuja funcionalidade pretende efetuar. 4. O sistema apresenta um menu adicional com os campos a preencher, relativos à funcionalidade escolhida no ponto 3. 5. O administrador preenche os campos e clica no botão para concluir a operação. 6. O sistema processa os dados recebidos e mostra uma mensagem de sucesso.
Fluxo Secundários	<ol style="list-style-type: none"> 5. A) O administrador pode fechar ou abrir outro menu de outra funcionalidade a qualquer momento. 7. A) Se ocorrer um erro, o sistema mostra uma mensagem do erro que ocorreu.
Exceções	
Pós-condição	
Casos de teste	

Tabela 18 - Descrição de caso de uso "Gerir Horários"

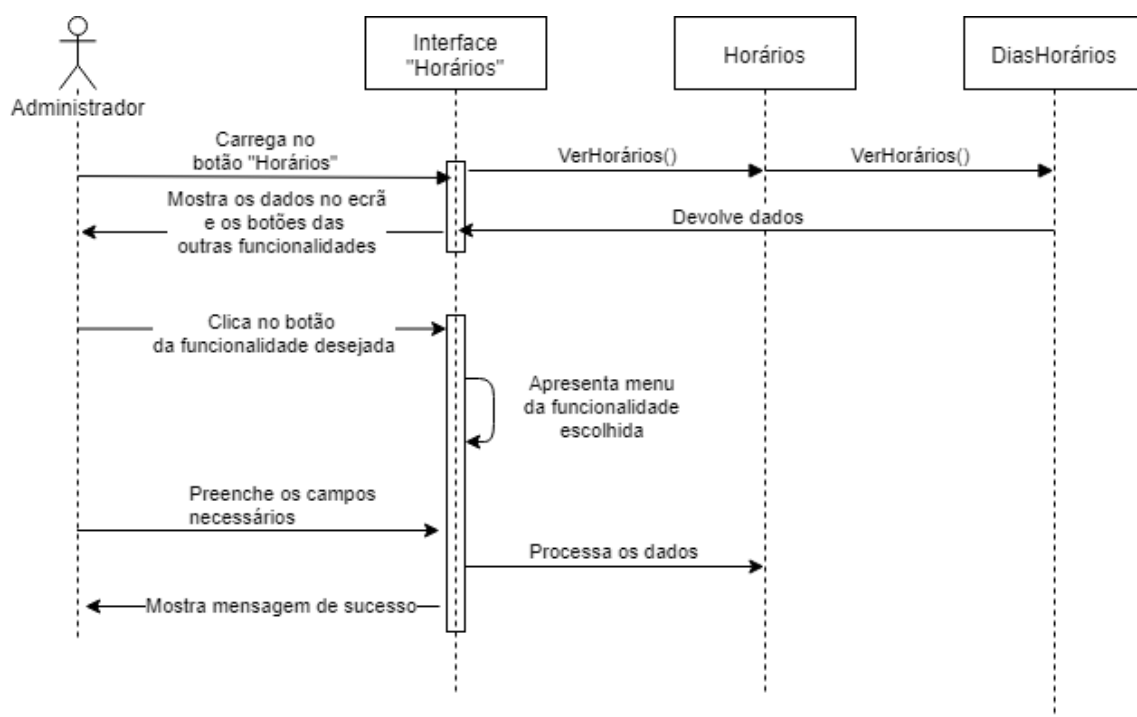


Figura 31 - Diagrama de sequência "Gerir Horários"

6.1.1.1.2. Colocar funcionários em horário

Nome	Colocar funcionários em horário
Objetivo	Atribuir um horário a vários funcionários
Atores	Administrador
Pré-condição	Login de administrador efetuado
Prioridade	Baixa
Fluxo Principal	<ol style="list-style-type: none"> 1. O administrador clica no botão “Atribuir horário” 2. O sistema mostra os vários horários da organização. 3. O administrador seleciona um horário. 4. O sistema apresenta os funcionários ativos e que não estejam nesse horário. 5. O administrador escolhe os funcionários a atribuir ao horário selecionado. 6. O sistema atribui o horário aos funcionários escolhidos. 7. O sistema mostra uma mensagem de sucesso.
Fluxo Secundários	<ol style="list-style-type: none"> 6. A) Se ocorrer um erro, o sistema mostra uma mensagem do erro que ocorreu.
Exceções	
Pós-condição	
Casos de teste	

Tabela 19 - Descrição de caso de uso "Colocar funcionários em horário"

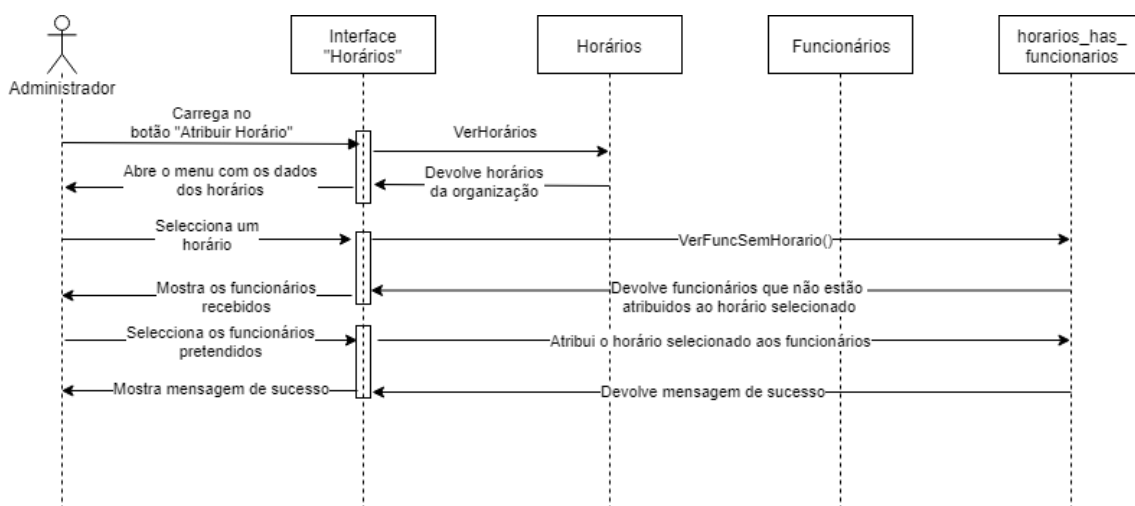


Figura 32 - Diagrama de sequência "Colocar Funcionário em Horário"